UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

DEPARTAMENTO DE INGENIERÍA DEL SOFTWARE E INTELIGENCIA ARTIFICIAL

ADAPTATIVE ROUTING PROTOCOLS FOR MOBILE AD HOC NETWORKS

PROTOCOLOS DE ENCAMINAMIENTO ADAPTATIVO PARA REDES MÓVILES AD HOC

TESIS DOCTORAL DE:
DELFÍN RUPÉREZ CAÑAS

DIRIGIDA POR:
LUIS JAVIER GARCÍA VILLALBA

Madrid, 2013

Adaptive Routing Protocols for Mobile Ad Hoc Networks

– –

Protocolos de Encaminamiento Adaptativo para Redes Móviles Ad Hoc

Thesis by

Delfín Rupérez Cañas

In Partial Fulfillment of the Requirements for the Degree of
Doctor por la Universidad Complutense de Madrid en el
Programa de Doctorado en Ingeniería Informática

Advisor

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, July 2013

Adaptive Routing Protocols for Mobile Ad Hoc Networks

Thesis by

Delfín Rupérez Cañas

In Partial Fulfillment of the Requirements for the Degree of
Doctor por la Universidad Complutense de Madrid en el
Programa de Doctorado en Ingeniería Informática

Advisor

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, July 2013

Protocolos de Encaminamiento Adaptativo para Redes Móviles Ad Hoc

TESIS DOCTORAL

Memoria presentada para obtener el título de
Doctor por la Universidad Complutense de Madrid
en el Programa de Doctorado en Ingeniería Informática

Delfín Rupérez Cañas

Dirigida por el profesor

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, Julio de 2013

Dissertation submitted by Delfín Rupérez Cañas to the *Departamento de Ingeniería del Software e Inteligencia Artificial* of the *Universidad Complutense de Madrid* in Partial Fulfillment of the Requirements for the Degree of *Doctor por la Universidad Complutense de Madrid en el Programa de Doctorado en Ingeniería Informática*.

Madrid, 2013.

(Submitted July 1, 2013)

Title:

Adaptive Routing Protocols
for Mobile Ad Hoc Networks

PhD Student:

Delfín Rupérez Cañas (delfinrc@fdi.ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
28040 Madrid, Spain

Advisor:

Luis Javier García Villalba (javiergv@fdi.ucm.es)

This work has been done within the Group of Analysis, Security and Systems (GASS, http://gass.ucm.es/en/), Research Group 910623 from the Universidad Complutense de Madrid (UCM) as part of the activities of the research projects of the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) FIT-360000-2007-48, TSI-020100-2008-365, TSI-020100-2009-374, TSI-020100-2010-482 and TSI-020100-2011-165. This research has also been supported by the Ministerio de Educación (MEC, Spain) through grant TME2009-00648, thanks to which part of this work was done during my stay in UK at University of Portsmouth (Faculty of Technology, School of Computing).

Tesis Doctoral presentada por el doctorando Delfín Rupérez Cañas en el Departamento de Ingeniería del Software e Inteligencia Artificial de la Universidad Complutense de Madrid para la obtención del título de Doctor por la Universidad Complutense de Madrid en el Programa de Doctorado en Ingeniería Informática.

Terminada en Madrid el 1 de julio de 2013.

Título:

Protocolos de Encaminamiento Adaptativo para Redes Móviles Ad Hoc

Doctorando:

Delfín Rupérez Cañas (delfinrc@fdi.ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
28040 Madrid, España

Director:

Luis Javier García Villalba (javiergv@fdi.ucm.es)

Esta tesis doctoral ha sido realizada dentro del grupo de investigación GASS (Grupo de Análisis, Seguridad y Sistemas, grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades de los proyectos de investigación del Ministerio de Industria, Turismo y Comercio (MITyC) *Semantic & Ambient Trust Technologies* (Programa PROFIT, referencia FIT-360000-2007-48), *Semantic & Ambient Trust Technologies II* (Subprograma Avanza I+D, referencia TSI-020100-2008-365), *Semantic & Ambient Trust Technologies III* (Subprograma Avanza I+D, referencia TSI-020100-2009-374), *Trust as a Service* (Subprograma Avanza Competitividad I+D+I, referencia TSI-020100-2010-482) y *Privacy-aware Accountability for a Trustworthy Future Internet* (Subprograma Avanza Competitividad I+D+I, referencia TSI-020100-2011-165). La presente investigación ha sido subvencionada por el Ministerio de Educación (MEC) con la Ayuda MEC TME2009-00648 para la obtención de la Mención Europea en el título de Doctor, gracias a la cual parte de esta investigación ha sido realizada en la *School of Computing* de la *Faculty of Technology* de la *University of Portsmouth* en Reino Unido.

This thesis is dedicated to my parents,

Antonio and Pilar,

who have taught me the true meaning of

courage, determination, perseverance, and love.

Esta tesis está dedicada a mis padres,

Antonio y Pilar,

que me han enseñado el verdadero significado de

coraje, determinación, perseverancia y amor.

Acknowledgments

I would like to express mi profound appreciation to my advisor, Javier García, for giving me the opportunity to work with him for the last five years. I am deeply indebted to him for providing a supportive environment for my research. His insight, patience, and encouragement have been invaluable to me during this process and have undoubtedly changed me for the better.

I would like to give thanks to my colleagues of the GASS research group for their assistance. Our numerous discussions and their insightful suggestions have greatly increased the quality, as well as the impact, of my PhD.

I must also give special thanks to all the people at IDSIA (Istituto Dalle Molle di Studi sull'Intelligenza Artificiale) for their invaluable scientific input.

I would also like to thank my friends for all the support I have received during this period.

Last, but surely not least, I am forever indebted to my parents for everything they have given me. Thank you for having been a wonderful model for me.

This work has been done within the Group of Analysis, Security and Systems (GASS, http://gass.ucm.es/en/), Research Group 910623 from the Universidad Complutense de Madrid (UCM) as part of the activities of the research projects of the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) FIT-360000-2007-48, TSI-020100-2008-365, TSI-020100-2009-374, TSI-020100-2010-482 and TSI-020100-2011-165. This research has also been supported by the Ministerio de Educación (MEC, Spain) through grant TME2009-00648, thanks to which part of this work was done during my stay in UK at University of Portsmouth (Faculty of Technology, School of Computing).

Agradecimientos

Quiero expresar mi más sincero agradecimiento a Javier García por darme la oportunidad de trabajar con él durante los últimos 5 años. Estoy profundamente en deuda con él por todo el apoyo proporcionado en este tiempo. Su visión, paciencia y aliento han sido fundamentales para mí durante este proceso y, sin duda, me han cambiado para mejor.

Quiero agradecer también muy especialmente a mis compañeros del Grupo de Análisis, Seguridad y Sistemas (GASS) por su ayuda. Nuestras numerosas discusiones y sus valiosas sugerencias han aumentado considerablemente la calidad y el impacto de esta investigación.

Quiero agradecer también muy especialmente a todos los investigadores del IDSIA por las inestimables discusiones mantenidas.

Asimismo, quiero agradecer a mis amigos por el ánimo y apoyo recibido.

Por último, pero ciertamente no menos importante, estoy eternamente agradecido a mis padres por todo lo que me han dado. Gracias por haber sido un modelo maravilloso para mí.

Esta tesis doctoral ha sido realizada dentro del grupo de investigación GASS (Grupo de Análisis, Seguridad y Sistemas, grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades de los proyectos de investigación del Ministerio de Industria, Turismo y Comercio (MITyC) *Semantic & Ambient Trust Technologies* (Programa PROFIT, referencia FIT-360000-2007-48), *Semantic & Ambient Trust Technologies II* (Subprograma Avanza I+D, referencia TSI-020100-2008-365), *Semantic & Ambient Trust Technologies III* (Subprograma Avanza I+D, referencia TSI-020100-2009-374), *Trust as a Service* (Subprograma Avanza Competitividad I+D+I, referencia TSI-020100-2010-482) y *Privacy-aware Accountability for a Trustworthy Future Internet* (Subprograma Avanza Competitividad I+D+I, referencia TSI-020100-2011-165).

La presente investigación ha sido subvencionada por el Ministerio de Educación (MEC) con la Ayuda MEC TME2009-00648 para la obtención de la Mención Europea en el título de Doctor, gracias a la cual parte de esta investigación ha sido realizada en la *School of Computing* de la *Faculty of Technology* de la *University of Portsmouth* en Reino Unido.

Abstract

A mobile ad hoc network (MANET) is a set of mobile nodes that communicate among themselves through wireless links. As opposed to conventional networks, a mobile ad hoc network does not need the existence of a previous infrastructure since each node relies on the others to communicate by creating the so called multi-hop communication. This type of networks has several drawbacks not found in conventional networks. For example, its topology can change quickly and unpredictably. Besides, variations in the capacities of nodes and connections may arise, as well as frequent errors in the transmission and a lack of security. Finally, the limited resources of nodes must be taken into account, since normally an ad-hoc network will contain devices fed by batteries.

The MANETs are dynamically built when a set of nodes creates paths in order to obtain connectivity among them. The nodes in a MANET may act not only as source or destination of a communication, but also as routers when a relationship between nodes cannot be achieved for reasons of reach. A routing protocol in a MANET requires providing a mechanism that maintains the routes towards the destinations given the movement of the nodes that may cause the destruction of the routes, and that it is necessary to find an alternative route in order to keep the communication between the nodes. Routing protocols for MANETs are often called protocols of level 2.5, since generally they are found above linking protocols like IEEE 802.11 and below the network IP protocol. In MANETs the conventional routing protocols will either have a very poor performance, or will not be applicable. As alternative, new routing algorithms are therefore needed, which can give adaptivity in an efficient and robust way.

Ant Colony Optimization (ACO) is a branch of Artificial Intelligence (AI) that uses the concepts of Swarm Intelligence and takes its inspiration from the behavior of ants in nature (bioinspired). ACO is applied to a wide range of different problems. Due to its properties of adaptivity and robustness, ACO has also attracted attention as a paradigm for routing in MANETs. ACO algorithms work in an iterative way. In each iteration, all artificial ants build a solution to the problem at hand in parallel, using the artificial pheromone matrix. Then, the pheromone matrix is updated based on the solutions that were found. This way, the pheromone matrix reflects information about good solutions that have been found so far, and allows ants in subsequent generations to use this information when building new solutions.

In this work an ACO routing protocol for mobile ad hoc networks based on AntHocNet is specified. As its predecessor, this new protocol, called AntOR, is hybrid in the sense that it contains elements from both reactive and proactive routing. Specifically, it combines a reactive route setup process with a proactive route maintenance and improvement process. Routing information is stored in pheromone tables that are similar to the ones used in other ACO routing algorithms. Forwarding of control and data packets is done in a stochastic way using these tables. Link failures are dealt with using specific reactive mechanisms, such as local route repair and the use of warning messages. Key aspects of the AntOR protocol are the disjoint-link and disjoint-node routes, separation between the regular pheromone and the virtual pheromone in the diffusion process and the exploration of routes, taking into consideration the number of hops in the best routes.

In this work a family of ACO routing protocols based on AntOR is also specified: Disjoint Node Routes (AntOR-DNR), Disjoint Link Routes (AntOR-DLR), Restrictive Disjoint Link Routes (AntOR-RDLR), Unicast Disjoint Link Routes (AntOR-UDLR), AntOR-v2 and HACOR. Somehow these protocols are based protocol successive refinements. AntOR-RDLR differs from AntOR-DLR in the pheromones updating process and

the route discovery mechanism. AntOR-UDLR consists in an unicast approach of AntOR-DLR that replace the notification messages of link failure, which are sent in broadcast mode in AntOR-DLR by unicast messages that are sent to the predecessor of the node reporting the link failure until they reach the source of the data session. AntOR-v2 and HACOR provide new optimization techniques as control packet buffering and outdated route management, and different management of link failures and route exploration.

In this work we present a parallelized version of AntOR that we call PAntOR. Using programming multiprocessor architectures based on shared memory protocol, PAntOR allows running tasks in parallel using threads. This parallelization is applicable in the route setup phase, route local repair process and link failure notification. In addition, a variant of PAntOR that consists of having more than one interface, that we call PAntoR-MI, is specified. This approach parallelizes the sending of broadcast messages by interface through threads.

The simulations were performed using the tool Network Simulator (NS-3). The results show that in most cases this new set of ACO routing protocols have lower overhead and higher delivered packet ratio than AntHocNet and AODV, being HACOR the best sequential approach. The results also indicate that the parallel approaches perform better that sequential approaches, with emphasis on the metric of average end-to-end delay, *jitter* and delivered packet ratio. We also show that PAntOR-MI is the most suitable for highly dynamic environments.

Keywords: Ant Colony Optimization, ACO, Bioinspired, Mobile Ad Hoc Networks, MANET, Routing Protocol, Swarm Intelligence.

Resumen

Una red móvil ad hoc es un conjunto de nodos móviles que se comunican entre sí a través de enlaces inalámbricos. A diferencia de las redes convencionales, una red móvil ad hoc no necesita la existencia de una infraestructura previa ya que cada nodo se apoya en los demás para conseguir comunicarse con otro creando la llamada comunicación multisalto. Este tipo de redes tiene varios inconvenientes que una red convencional no presenta. La topología de este tipo de redes puede cambiar rápidamente y de una forma impredecible. Además, pueden surgir variaciones en las capacidades de los nodos y enlaces, errores frecuentes en la transmisión y falta de seguridad. Por último, se deben tener en cuenta los recursos limitados de los nodos ya que normalmente una red ad hoc estará formada por dispositivos alimentados por baterías.

Las redes móviles ad hoc se construyen de forma dinámica cuando un conjunto de nodos crean rutas entre sí para conseguir la conectividad entre ellos. Los nodos de la red móvil ad hoc pueden actuar como origen o destino de una comunicación, pero también como encaminadores cuando una relación entre nodos no se puede realizar directamente por motivos de alcance. Un protocolo de encaminamiento de una red móvil ad hoc necesita proveer un mecanismo que mantenga las rutas hacia los destinos frente al movimiento de los nodos que puede provocar que las rutas se destruyan, y sea necesario encontrar una ruta alternativa para mantener la comunicación entre los nodos. Con frecuencia se les denomina de nivel 2.5, ya que es habitual encontrarlos por encima de protocolos de enlace como IEEE 802.11 y por debajo del protocolo de red IP. En redes móviles ad hoc los protocolos de encaminamiento convencionales o bien tendrán un rendimiento muy pobre, o bien serán simplemente inaplicables. Como alternativa se desarrollan protocolos específicos de encaminamiento, que logran la adaptabilidad de una forma eficiente y robusta.

El Algoritmo de Optimización de la Colonia de Hormigas, abreviadamente ACO, es una rama de la Inteligencia Artificial que usa conceptos de inteligencia colectiva y que se inspira en el comportamiento de las hormigas en la naturaleza. ACO se aplica a una amplia gama de problemas diferentes. Debido a sus propiedades de adaptabilidad y robustez, también se ha convertido en un paradigma para el encaminamiento en redes móviles ad hoc. Los algoritmos ACO trabajan de forma iterativa. En cada paso las hormigas artificiales construyen en paralelo una solución para el problema en cuestión, utilizando la matriz de feromona artificial. A continuación, se actualiza la matriz de feromona sobre la base de las soluciones encontradas. De esta manera, la matriz de feromona refleja información sobre las buenas soluciones que se han encontrado hasta la fecha, y permite a las hormigas de las generaciones posteriores utilizar esta información para crear otras nuevas.

En este trabajo se especifica un protocolo de encaminamiento ACO para redes móviles ad hoc basado en AntHocNet. Como su predecesor, este nuevo protocolo llamado AntOR es híbrido, en el sentido de que contiene elementos de encaminamiento tanto reactivos como proactivos. En concreto, combina un proceso reactivo de establecimiento de ruta con un proceso proactivo de mantenimiento y exploración de nuevas rutas. La información de encaminamiento se almacena en tablas de feromona que son similares a las utilizadas por otros algoritmos de encaminamiento ACO. El reenvío de paquetes de datos y de control se realiza de una manera estocástica con el uso de estas tablas. Los fallos de enlace se tratan con mecanismos reactivos específicos, tales como la reparación local de ruta y el uso de mensajes de aviso. Los aspectos clave del protocolo AntOR son la utilización de rutas disjuntas de nodo y disjuntas de enlace, la separación entre la feromona regular y la feromona virtual en el proceso de difusión y la exploración de nuevas rutas, que tiene en cuenta el número de saltos en las mejores rutas.

En este trabajo también se especifica una familia de protocolos de encaminamiento ACO basada en AntOR: AntOR disjunto de nodo (AntOR-DNR), AntOR disjunto de enlace (AntOR-DLR), AntOR restrictivo (AntOR-RDLR), AntOR unicast (AntOR-UDLR), AntOR-v2 y HACOR. Todos estos protocolos son refinamientos sucesivos del protocolo original. AntOR-RDLR difiere de AntOR-DLR en el proceso de actualización de feromonas y en el mecanismo de descubrimiento de rutas. AntOR-UDLR es una aproximación unicast de AntOR-DLR, que sustituye los mensajes de notificación de fallo de enlace, que se envían en modo difusión (broadcast) en AntOR-DLR, por mensajes unicast que se envían al predecesor del nodo que informa del fallo del enlace hasta llegar a la fuente de la sesión de datos. AntOR-v2 y HACOR proporcionan nuevas técnicas de optimización como almacenamiento de paquetes de control y gestión de rutas obsoletas, y diferentes gestiones de fallos de enlace y de exploración de rutas.

En este trabajo también se especifica una versión paralelizada de AntOR, PAntOR, que hace uso de arquitecturas multiprocesador de programación basado en protocolo de memoria compartida y permite ejecutar tareas en paralelo usando hilos, siendo aplicable esta paralelización en la fase de establecimiento de ruta, en el proceso de reparación local de rutas y en la notificación de fallos de enlace. Asimismo, se especifica una variante con más de una interfaz (PAntOR-MI), que paraleliza el envío de mensajes de difusión por interfaz a través de hilos.

Las simulaciones se realizaron utilizando la herramienta Network Simulator (NS-3). Los resultados muestran que en la mayoría de los casos este nuevo conjunto de protocolos de encaminamiento ACO tiene menor sobrecarga y mayor tasa de entrega de paquetes que AntHocNet y AODV, siendo HACOR la mejor variante secuencial. Los resultados también indican que las aproximaciones paralelas funcionan mejor que las secuenciales, especialmente en lo relativo al retardo extremo a extremo, al *jitter* y a la tasa de entrega de paquetes, y que PAntOR-MI es el más adecuado en ambientes altamente dinámicos.

Palabras clave: Algoritmo de Optimización de la Colonia de Hormigas, Bioinspirado, Inteligencia Colectiva, Protocolo de Encaminamiento, Redes Móviles Ad Hoc.

Contents

Contents xxiii

List of Figures xxxi

List of Tables xxxiii

List of Algorithms xxxv

List of Acronyms xl

I Description of the Research xli

1 Introduction 1
- 1.1 Objetives . . . 2
- 1.2 Problem Identification . . . 2
- 1.3 Research Context . . . 2
- 1.4 Summary of Contributions . . . 3
- 1.5 Outline of the Thesis . . . 4

2 Mobile Ad Hoc Networks 5
- 2.1 Evolution Historical . . . 5
- 2.2 Characteristics . . . 6
- 2.3 Standard IEEE 802.11 . . . 7
- 2.4 Classification . . . 9
- 2.5 Applications . . . 9
- 2.6 Summary . . . 11

3 Routing in Mobile Ad Hoc Networks 13
- 3.1 Routing Protocols . . . 13
- 3.2 Classification of Routing Protocols . . . 15
- 3.3 OLSR Protocol . . . 17
  - 3.3.1 Protocol Functioning . . . 17
  - 3.3.2 OLSR Packet Format . . . 18
    - 3.3.2.1 Packet Header . . . 19
    - 3.3.2.2 Message Header . . . 19
  - 3.3.3 MID Message . . . 20
  - 3.3.4 Hello Message . . . 20
    - 3.3.4.1 Hello Message Format . . . 21
    - 3.3.4.2 Hello Message Processing . . . 22

3.3.5 Neighbour Discovery . . . 22
3.3.5.1 Detection of Connections at Link Level . . . 22
3.3.5.2 Neighbour Detection . . . 22
3.3.6 Multipoint Relay (MPR) . . . 23
3.3.6.1 MPR Selection . . . 24
3.3.7 Topology Discovery in OLSR . . . 24
3.3.7.1 Functioning . . . 24
3.3.7.2 TC Message Format . . . 25
3.3.8 Routing Table Calculation . . . 25
3.4 AODV Protocol . . . 26
3.4.1 Control Messages . . . 26
3.4.1.1 RREQ Message . . . 26
3.4.1.2 RREP Message . . . 27
3.4.1.3 RERR Message . . . 28
3.4.1.4 Hello Message . . . 29
3.4.2 Route Discovery . . . 29
3.4.3 Route Maintenance . . . 29
3.5 ZRP . . . 30
3.6 Summary . . . 30

4 Ant Colony Optimization (ACO) Algorithm 33
4.1 Introduction . . . 33
4.2 Double Bridge Experiment . . . 34
4.3 Artificial Ants . . . 34
4.4 Simple Ant Colony Optimization (S-ACO) . . . 36
4.4.1 Functioning Modes . . . 37
4.4.2 Path Search . . . 37
4.4.3 Path Retracing and Pheromone Update . . . 37
4.4.4 Pheromone Trail Evaporation . . . 38
4.4.5 ACO Metaheuristic . . . 38
4.5 Parallel Approach . . . 39
4.6 Summary . . . 40

5 Adaptive Routing 41
5.1 Introduction . . . 41
5.2 ACO Routing . . . 41
5.3 ACO Routing in Mobile Ad hoc Networks . . . 42
5.4 Related Work . . . 42
5.5 Summary . . . 49

6 Adaptive Routing Protocols for Mobile Ad Hoc Networks 51
6.1 Ant Optimized Routing (AntOR): Overview . . . 51
6.2 Ant Optimized Routing (AntOR): Data Structures . . . 52
6.2.1 Routing Table . . . 52
6.2.2 Neighbor Table . . . 54
6.3 Ant Optimized Routing (AntOR): Functioning . . . 55
6.3.1 Route Setup . . . 55
6.3.1.1 Reactive Forward Process . . . 55
6.3.1.2 Reactive Backward Process . . . 57
6.3.2 Data Stochastic Routing . . . 60

6.3.3 Established Path Maintenance and Exploration of New Routes . . . 60
6.3.3.1 Pheromone Diffusion . . . 61
6.3.3.2 Ant Proactive Sending . . . 63
6.3.3.3 Disjoint Link / Node Routes . . . 63
6.3.3.4 Functioning . . . 66
6.3.4 Management of Link Failures . . . 67
6.3.5 Summary . . . 69
6.4 AntOR - Disjoint Link Route (AntOR-DLR) . . . 72
6.5 AntOR - Disjoint Node Route (AntOR-DNR) . . . 75
6.6 AntOR - Restrictive Disjoint Link Route (AntOR-RDLR) . . . 77
6.7 AntOR - Unicast Disjoint Link Route (AntOR-UDLR) . . . 79
6.8 AntOR-v2 . . . 82
6.9 Hybrid ACO Routing (HACOR) . . . 83
6.10 Parallel AntOR (PAntOR) . . . 86
6.11 PAntOR - Multiple Interface (PAntOR-MI) . . . 88
6.12 Summary . . . 90

7 Simulations and Results 91
7.1 Election of Network Simulator . . . 91
7.2 Simulation Environment . . . 92
7.3 Performance Metrics . . . 105
7.4 Evaluation of AntOR-DLR Protocol . . . 105
7.4.1 Throughput . . . 105
7.4.2 Delivered Data Packet Ratio . . . 106
7.4.3 Average End-to-End Delay . . . 106
7.4.4 Overhead in Number of Packets . . . 107
7.4.5 Overhead in Number of Bytes . . . 107
7.5 Evaluation of AntOR-DNR Protocol . . . 108
7.5.1 Delivered Data Packet Ratio . . . 108
7.5.2 Average End-to-End Delay . . . 109
7.5.3 Jitter . . . 109
7.6 Evaluation of AntOR-RDLR Protocol . . . 110
7.6.1 Setting of MAX_HOP . . . 110
7.6.2 Throughput . . . 111
7.6.3 Delivered Data Packet Ratio . . . 112
7.7 Evaluation of AntOR-UDLR Protocol . . . 112
7.7.1 Throughput . . . 112
7.7.2 Delivered Data Packet Ratio . . . 113
7.7.3 Average End-to-End Delay . . . 113
7.7.4 Overhead in the Number of Packets . . . 115
7.7.5 Overhead in the Number of Bytes . . . 115
7.8 Evaluation of AntOR-v2 Protocol . . . 116
7.8.1 Throughput . . . 116
7.8.2 Delivered Data Packet Ratio . . . 117
7.8.3 Average End-to-End Delay . . . 118
7.8.4 Jitter . . . 118
7.8.5 Overhead in Number of Packets . . . 119
7.8.6 Overhead in Number of Bytes . . . 120
7.9 Evaluation of HACOR Protocol . . . 121

7.9.1 Throughput . . . 121
7.9.2 Delivered Data Packet Ratio . . . 122
7.9.3 Average End-to-End Delay . . . 123
7.9.4 Jitter . . . 124
7.9.5 Overhead in Number of Packets . . . 125
7.9.6 Overhead in Number of Bytes . . . 126
7.10 Evaluation of PAntOR Protocol . . . 127
7.10.1 Throughput . . . 127
7.10.2 Delivered Data Packet Ratio . . . 128
7.10.3 Average End-to-End Delay . . . 129
7.10.4 Jitter . . . 130
7.10.5 Overhead in Number of Packets . . . 131
7.11 Evaluation of PAntOR-MI Protocol . . . 131
7.11.1 Delivered Data Packet Ratio . . . 132
7.12 Summary . . . 132

8 Concluding Remarks and Future Work **135**
8.1 Future Work . . . 137

Bibliography **139**

II Resumen de la Investigación **147**

9 Introducción **151**
9.1 Objetivos . . . 152
9.2 Identificación del Problema . . . 152
9.3 Contexto de la Investigación . . . 152
9.4 Resumen de la Tesis . . . 153
9.5 Estructura de la Tesis . . . 154

10 Redes Móviles Ad Hoc **155**
10.1 Evolución Histórica . . . 155
10.2 Características . . . 156
10.3 Estándar IEEE 802.11 . . . 158
10.4 Clasificación . . . 159
10.5 Aplicaciones . . . 160
10.6 Resumen . . . 161

11 Encaminamiento en Redes Móviles Ad Hoc **163**
11.1 Protocolos de Encaminamiento . . . 163
11.2 Clasificación de los Protocolos de Encaminamiento . . . 165
11.3 Protocolo OLSR . . . 167
11.3.1 Funcionamiento del Protocolo . . . 168
11.3.2 Formato del Paquete OLSR . . . 169
11.3.2.1 Cabecera del Paquete . . . 169
11.3.2.2 Cabecera del Mensaje . . . 170
11.3.3 Mensaje MID . . . 170
11.3.4 Mensaje **Hello** . . . 171
11.3.4.1 Formato del Mensaje **Hello** . . . 171

11.3.4.2 Procesamiento del Mensaje *Hello* . . . 172
11.3.5 Descubrimiento de Vecinos . . . 172
11.3.5.1 Detección de Conexiones a Nivel de Enlace . . . 172
11.3.5.2 Detección de Vecinos . . . 173
11.3.6 Multipuntos de Retransmisión (MPR) . . . 173
11.3.6.1 Selección de MPR . . . 174
11.3.7 Descubrimiento de la Topología en OLSR . . . 175
11.3.7.1 Funcionamiento . . . 175
11.3.7.2 Formato de los Mensajes TC . . . 175
11.3.8 Cálculo de las Tablas de Rutas . . . 176
11.4 Protocolo AODV . . . 176
11.4.1 Mensajes de Control . . . 176
11.4.1.1 Mensajes RREQ . . . 177
11.4.1.2 Mensajes RREP . . . 178
11.4.1.3 Mensajes RERR . . . 179
11.4.1.4 Mensajes *Hello* . . . 179
11.4.2 Descubrimiento de Rutas . . . 180
11.4.3 Mantenimiento de Rutas . . . 180
11.5 ZRP . . . 181
11.6 Resumen . . . 181

12 El Algoritmo Ant Colony Optimization (ACO) 183
12.1 Introducción . . . 183
12.2 Experimento del Doble Puente . . . 184
12.3 Hormigas Artificiales . . . 185
12.4 Simple Ant Colony Optimization (S-ACO) . . . 186
12.4.1 Modos de funcionamiento . . . 187
12.4.2 Búsqueda de Caminos . . . 187
12.4.3 Trazado de Ruta y Actualización de Feromona . . . 188
12.4.4 Evaporación de las Marcas de Feromona . . . 188
12.4.5 Meta-Heurística ACO . . . 189
12.5 Aproximación Paralela . . . 190
12.6 Resumen . . . 190

13 Encaminamiento Adaptativo 193
13.1 Introducción . . . 193
13.2 Encaminamiento ACO . . . 193
13.3 Encaminamiento ACO en Redes Móviles Ad Hoc . . . 194
13.4 Trabajos Relacionados . . . 194
13.5 Resumen . . . 202

14 Protocolos de Encaminamiento Adaptativo para Redes Móviles Ad Hoc 203
14.1 Ant Optimized Routing (AntOR): Generalidades . . . 203
14.2 Ant Optimized Routing (AntOR): Estructuras de Datos . . . 204
14.2.1 Tabla de Encaminamiento . . . 204
14.2.2 Tabla de Vecinos . . . 206
14.3 Ant Optimized Routing (AntOR): Funcionamiento . . . 207
14.3.1 Establecimiento de Ruta . . . 207
14.3.1.1 Proceso Reactivo Hacia Adelante . . . 207
14.3.1.2 Proceso Reactivo Hacia Atrás . . . 210

14.3.2 Encaminamiento Estocástico de los Datos . . . 213
14.3.3 Mantenimiento de Rutas Establecidas y Exploración de Nuevas Rutas 213
14.3.3.1 Difusión de Feromona . . . 213
14.3.3.2 Envío Proactivo de Hormigas . . . 215
14.3.3.3 Rutas Disjuntas de Enlace/Nodo . . . 215
14.3.3.4 Funcionamiento . . . 218
14.3.4 Gestión de Fallos de Enlace . . . 220
14.3.5 Resumen . . . 222
14.4 AntOR - Disjoint Link Route (AntOR-DLR) . . . 225
14.5 AntOR - Disjoint Node Route (AntOR-DNR) . . . 228
14.6 AntOR - Restrictive Disjoint Link Route (AntOR-RDLR) . . . 230
14.7 AntOR - Unicast Disjoint Link Route (AntOR-UDLR) . . . 232
14.8 AntOR-v2 . . . 235
14.9 Hybrid ACO Routing (HACOR) . . . 237
14.10 Parallel AntOR (PAntOR) . . . 239
14.11 PAntOR - Multiple Interface (PAntOR-MI) . . . 242
14.12 Resumen . . . 243

15 Simulaciones y Resultados **245**
15.1 Elección del Simulador de Redes . . . 245
15.2 Entorno de Simulación . . . 246
15.3 Métricas de Rendimiento . . . 259
15.4 Evaluación del Protocolo AntOR-DLR . . . 259
15.4.1 Throughput . . . 259
15.4.2 Ratio de Entrega de Paquetes de Datos . . . 260
15.4.3 Retardo Medio Extremo a Extremo . . . 260
15.4.4 Sobrecarga en el Número de Paquetes . . . 261
15.4.5 Sobrecarga en el Número de Bytes . . . 262
15.5 Evaluación del Protocolo AntOR-DNR . . . 262
15.5.1 Ratio de Entrega de Paquetes de Datos . . . 262
15.5.2 Retardo Medio Extremo a Extremo . . . 263
15.5.3 Jitter . . . 263
15.6 Evaluación del Protocolo AntOR-RDLR . . . 264
15.6.1 Ajuste de MAX_HOP . . . 264
15.6.2 Throughput . . . 265
15.6.3 Ratio de Entrega de Paquetes de Datos . . . 266
15.7 Evaluación del Protocolo AntOR-UDLR . . . 266
15.7.1 Throughput . . . 266
15.7.2 Ratio de Entrega de Paquetes de Datos . . . 267
15.7.3 Retardo medio extremo a extremo . . . 267
15.7.4 Sobrecarga en el Número de Paquetes . . . 269
15.7.5 Sobrecarga en el Número de Bytes . . . 269
15.8 Evaluación del Protocolo AntOR-v2 . . . 270
15.8.1 Throughput . . . 270
15.8.2 Ratio de Entrega de Paquetes de Datos . . . 271
15.8.3 Retardo Medio Extremo a Extremo . . . 272
15.8.4 Jitter . . . 272
15.8.5 Sobrecarga en el Número de Paquetes . . . 273
15.8.6 Sobrecarga en el Número de Bytes . . . 274

15.9 Evaluación del Protocolo HACOR . . . 275
15.9.1 Throughput . . . 275
15.9.2 Ratio de Entrega de Paquetes de Datos . . . 276
15.9.3 Retardo Medio Extremo a Extremo . . . 277
15.9.4 Jitter . . . 278
15.9.5 Sobrecarga en el Número de Paquetes . . . 279
15.9.6 Sobrecarga en el Número de Bytes . . . 280
15.10 Evaluación del Protocolo PAntOR . . . 281
15.10.1 Throughput . . . 282
15.10.2 Ratio de Entrega de Paquetes de Datos . . . 282
15.10.3 Retardo Medio Extremo a Extremo . . . 283
15.10.4 Jitter . . . 284
15.10.5 Sobrecarga en el Número de Paquetes . . . 285
15.11 Evaluación del Protocolo PAntOR-MI . . . 285
15.11.1 Ratio de Entrega de Paquetes de Datos . . . 286
15.12 Resumen . . . 286

16 Conclusiones y Trabajo Futuro **289**
16.1 Trabajos Futuros . . . 291

III Papers Related to This Thesis **293**

A List of Publications **295**
A.1 Bio-Inspired Routing Protocol for Mobile Ad Hoc Networks . . . 297
A.2 Secure Extension to the Optimised Link State Routing Protocol . . . 307
A.3 An Extension Proposal of AntOR for Parallel Computing . . . 315
A.4 A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks . . . 321
A.5 Comparing AntOR-Disjoint Node Routing Protocol with Its Parallel Extension . . . 327
A.6 Immune Systems for ACO-Based Routing Optimization . . . 333
A.7 ANTOR-UDLR: Aproximación Unicast de un Protocolo de Encaminamiento para Redes Móviles Ad Hoc . . . 339
A.8 Multiple Interface Parallel Approach of Bioinspired Routing Protocol for Mobile Ad Hoc Networks . . . 345
A.9 Restrictive Disjoint-Link-Based Bioinspired Routing Protocol for Mobile Ad Hoc Networks . . . 351
A.10 Technique to Neutralize Link Failures for an ACO-Based Routing Algorithm 357
A.11 Parallel Approach of a Bioinspired Routing Protocol for MANETs . . . 367
A.12 Adaptive Routing Protocol for Mobile Ad Hoc Networks . . . 373
A.13 An Ant-Based Adaptive Distributed Routing Protocol for Mobile Ad Hoc Networks . . . 385
A.14 HACOR: Hybrid ACO Routing Protocol for Mobile Ad Hoc Networks . . . 389
A.15 Routing Techniques Based on Swarm Intelligence . . . 397

List of Figures

1.1 Evolution scheme of AntOR . . . 3

2.1 Mobile ad hoc network . . . 8

3.1 Taxonomy of routing protocols in mobile ad hoc networks . . . 15
3.2 Formato del paquete OLSR . . . 19
3.3 OLSR packet header . . . 19
3.4 Message header . . . 20
3.5 MID message format . . . 21
3.6 Hello message format . . . 21
3.7 Selection process of MPR nodes . . . 23
3.8 Difference between pure flooding and the use of MPR nodes . . . 24
3.9 TC message format . . . 25
3.10 RREQ message format in AODV . . . 26
3.11 Propagation of a RREQ message in AODV . . . 27
3.12 RREP message format in AODV . . . 27
3.13 Back path of RREP message to origin . . . 28
3.14 RERR format message in AODV . . . 28

6.1 Updating scenario of routing table (AntOR) . . . 53
6.2 Scenario that represents the neighborhood of node A (AntOR) . . . 54
6.3 Example of the selection of an individual . . . 56
6.4 First route setup process (AntOR) . . . 57
6.5 Example of pheromone settlement (AntOR) . . . 59
6.6 Functioning of route setup process (AntOR) . . . 60
6.7 Example of selection of the best pheromone value (AntOR) . . . 62
6.8 Comparison of the two modalities (AntOR) . . . 64
6.9 Disjoint link routes a) versus non-disjoint b) (AntOR) . . . 64
6.10 Disjoint node routes a) versus non-disjoint b) (AntOR). . . . 65
6.11 Scenario: Disjoint link route (AntOR) . . . 65
6.12 Scenario: Disjoint node route (AntOR) . . . 65
6.13 Example of route exploration (AntOR) . . . 67
6.14 Exploration scheme using the *distance* metric (AntOR) . . . 67
6.15 Link failure management (AntOR) . . . 68
6.16 Example of local route repair (AntOR) . . . 69
6.17 Example of marking and route settlement (AntOR) . . . 70
6.18 Scenario example of protocol functioning (AntOR) . . . 70
6.19 Scheme of routing diffusion (AntOR) . . . 71
6.20 Functioning scheme of AntOR . . . 72
6.21 Representative scheme of disjoint link routes (AntOR-DLR) . . . 73

6.22 Flowchart of disjoint link routes (AntOR-DLR) . . . 74
6.23 Example I: One data session (AntOR-DLR) . . . 75
6.24 Example II: Two data sessions (AntOR-DLR) . . . 75
6.25 Representative scheme of disjoint node routes (AntOR-DNR) . . . 76
6.26 Flowchart of disjoint node routes (AntOR-DNR) . . . 76
6.27 Example of proactive process (AntOR-RDLR) . . . 78
6.28 Link Failure Managemente (AntOR-UDLR) . . . 79
6.29 Unicast Link Notification format message (AntOR-UDLR) . . . 80
6.30 Example of link failure management - case b (AntOR-UDLR) . . . 81
6.31 Example of link failure management - case c (AntOR-UDLR) . . . 82
6.32 Example of proactive process (AntOR-v2) . . . 83
6.33 Loop elimination process (HACOR) . . . 85
6.34 Example of path exploration (HACOR) . . . 85
6.35 Parallelization of route setup process (PAntOR) . . . 87
6.36 Parallelization of local route repair process (PAntOR) . . . 87
6.37 Parallelization of link failure notification process (P-AntOR) . . . 88
6.38 Example of functioning (PAntOR) . . . 88
6.39 Functioning PAntOR-MI . . . 89

7.1 Throughput (AntOR-DLR) . . . 106
7.2 Delivered data packet ratio (AntOR-DLR) . . . 106
7.3 Average end-to-end delay (AntOR-DLR) . . . 107
7.4 Overhead in number of packets (AntOR-DLR) . . . 107
7.5 Overhead in number of bytes (AntOR-DLR) . . . 108
7.6 Delivered data packet ratio (AntOR-DNR) . . . 109
7.7 Average end-to-end delay (AntOR-DNR) . . . 109
7.8 Jitter (AntOR-DNR) . . . 110
7.9 Setting of MAX_HOP - case a (AntOR-RDLR) . . . 110
7.10 Setting of MAX_HOP - case b (AntOR-RDLR) . . . 111
7.11 Throughput (AntOR-RDLR) . . . 111
7.12 Delivered data packet ratio (AntOR-RDLR) . . . 112
7.13 Throughput - case a (AntOR-UDLR) . . . 113
7.14 Throughput - case b (AntOR-UDLR) . . . 113
7.15 Delivered data packet ratio - case a (AntOR-UDLR) . . . 114
7.16 Delivered data packet ratio - case b (AntOR-UDLR) . . . 114
7.17 Average end-to-end delay - case a (AntOR-UDLR) . . . 114
7.18 Average end-to-end delay - case b (AntOR-UDLR) . . . 115
7.19 Overhead in the Number of Packets (AntOR-UDLR) . . . 115
7.20 Overhead in the Number of Bytes (AntOR-UDLR) . . . 116
7.21 Throughput (AntOR-v2) . . . 117
7.22 Delivered data packet ratio - case a (AntOR-v2) . . . 117
7.23 Delivered data packet ratio - case b (AntOR-v2) . . . 118
7.24 Average end-to-end delay (AntOR-v2) . . . 118
7.25 Jitter - case a (AntOR-v2) . . . 119
7.26 Jitter - case b (AntOR-v2) . . . 119
7.27 Overhead in number of packets - case a (AntOR-v2) . . . 120
7.28 Overhead in number of packets - case b (AntOR-v2) . . . 120
7.29 Overhead in number of bytes (AntOR-v2) . . . 121
7.30 Throughput - case a (HACOR) . . . 122
7.31 Throughput - case b (HACOR) . . . 122

7.32 Delivered data packet ratio - case a (HACOR) . . . 123
7.33 Delivered data packet ratio - case b (HACOR) . . . 123
7.34 Average end-to-end delay - case a (HACOR) . . . 124
7.35 Average end-to-end delay - case b (HACOR) . . . 124
7.36 Jitter - case a (HACOR) . . . 125
7.37 Jitter - case b (HACOR) . . . 125
7.38 Overhead in number of packets - case a (HACOR) . . . 126
7.39 Overhead in number of packets - case b (HACOR) . . . 126
7.40 Overhead in number of bytes - case a (HACOR) . . . 127
7.41 Overhead in number of bytes - case b (HACOR) . . . 127
7.42 Throughput (PAntOR) . . . 128
7.43 Delivered data packet ratio - case a (PAntOR) . . . 128
7.44 Delivered data packet ratio - case b (PAntOR) . . . 129
7.45 Average end-to-end delay - case a (PAntOR) . . . 129
7.46 Average end-to-end delay - case b (PAntOR) . . . 130
7.47 Jitter - case a (PAntOR) . . . 130
7.48 Jitter - case b (PAntOR) . . . 131
7.49 Overhead in number of packets (PAntOR) . . . 131
7.50 Delivered data packet ratio (PAntOR-MI) . . . 132

9.1 Esquema evolutivo de AntOR . . . 153

10.1 Red móvil ad hoc . . . 158

11.1 Taxonomía de protocolos de encaminamiento en redes móviles ad hoc . . . 165
11.2 Formato del paquete OLSR . . . 169
11.3 Cabecera del paquete OLSR . . . 169
11.4 Cabecera del mensaje . . . 170
11.5 Formato del mensaje MID . . . 171
11.6 Formato del mensaje **Hello** . . . 171
11.7 Proceso de selección de nodos MPR . . . 174
11.8 Diferencia entre la difusión pura y la difusión usando nodos MPR . . . 174
11.9 Formato del mensaje TC . . . 175
11.10 Formato del mensaje RREQ de AODV . . . 177
11.11 Propagación del un mensaje RREQ en AODV . . . 177
11.12 Formato del mensaje RREP de AODV . . . 178
11.13 Camino del mensaje de vuelta RREP al origen . . . 179
11.14 Formato del mensaje RERR de AODV . . . 179

14.1 Escenario actualización de la tabla de encaminamiento (AntOR) . . . 206
14.2 Escenario que representa la vecindad del nodo A (AntOR) . . . 207
14.3 Ejemplo de selección de un individuo . . . 209
14.4 Proceso del primer establecimiento de ruta (AntOR) . . . 209
14.5 Ejemplo de asentamiento de la feromona (AntOR) . . . 212
14.6 Funcionamiento del proceso de establecimiento de ruta (AntOR) . . . 212
14.7 Ejemplo de selección del mejor valor de feromona (AntOR) . . . 215
14.8 Comparativa de las dos modalidades (AntOR) . . . 216
14.9 Rutas de enlaces disjuntos a) versus no disjuntos b) (AntOR) . . . 217
14.10 Rutas de nodos disjuntos a) versus no disjuntos b) (AntOR). . . . 217
14.11 Escenario: Ruta de enlace disjunto (AntOR) . . . 217

14.12 Escenario: Ruta de nodo disjunto (AntOR) . . . 218
14.13 Ejemplo de exploración de rutas (AntOR) . . . 219
14.14 Esquema de exploración usando la métrica *distancia* (AntOR) . . . 220
14.15 Gestión fallos de enlace (AntOR) . . . 221
14.16 Ejemplo de reparación local de ruta (AntOR) . . . 222
14.17 Ejemplo de marcado y asentamiento de ruta (AntOR) . . . 222
14.18 Ejemplo de escenario del funcionamiento del protocolo (AntOR) . . . 223
14.19 Esquema de difusión de encaminamiento (AntOR) . . . 224
14.20 Diagrama funcionamiento de AntOR . . . 225
14.21 Esquema representativo de rutas disjuntas de enlace (AntOR-DLR) . . . 226
14.22 Diagrama flujo calculo de rutas disjuntas de enlace (AntOR-DLR) . . . 227
14.23 Ejemplo I: Una sesión de datos (AntOR-DLR) . . . 228
14.24 Ejemplo II: Dos sesiones de datos (AntOR-DLR) . . . 228
14.25 Esquema representativo de rutas disjuntas de nodo (AntOR-DNR) . . . 229
14.26 Diagrama de flujo de rutas de nodo disjunto (AntOR-DNR) . . . 229
14.27 Ejemplo en el proceso de proactivo (AntOR-RDLR) . . . 232
14.28 Gestión de Fallo de Enlace (AntOR-UDLR) . . . 233
14.29 Formato del mensaje *unicast* de notificación de enlace (AntOR-UDLR) . . . 234
14.30 Ejemplo de gestión de fallo de enlace - caso b (AntOR-UDLR) . . . 235
14.31 Ejemplo de gestión de fallo de enlace - caso c (AntOR-UDLR) . . . 235
14.32 Ejemplo de proceso proactivo (AntOR-v2) . . . 237
14.33 Proceso eliminación de bucles (HACOR) . . . 238
14.34 Ejemplo de exploración de caminos (HACOR) . . . 239
14.35 Paralelización del proceso de establecimiento de ruta (PAntOR) . . . 240
14.36 Paralelización del proceso de reparación local de ruta (PAntOR) . . . 241
14.37 Paralelización del proceso de notificación de fallo de enlace (P-AntOR) . . . 241
14.38 Ejemplo del funcionamiento (PAntOR) . . . 242
14.39 Funcionamiento PAntOR-MI . . . 243

15.1 Throughput (AntOR-DLR) . . . 260
15.2 Ratio de entrega de paquetes de datos (AntOR-DLR) . . . 260
15.3 Retardo medio extremo a extremo (AntOR-DLR) . . . 261
15.4 Sobrecarga en el número de paquetes (AntOR-DLR) . . . 261
15.5 Sobrecarga en el número de bytes (AntOR-DLR) . . . 262
15.6 Ratio de entrega de paquetes de datos (AntOR-DNR) . . . 263
15.7 Retardo medio extremo a extremo (AntOR-DNR) . . . 263
15.8 Jitter (AntOR-DNR) . . . 264
15.9 Ajuste de MAX_HOP - caso a (AntOR-RDLR) . . . 264
15.10 Ajuste de MAX_HOP - caso b (AntOR-RDLR) . . . 265
15.11 Throughput (AntOR-RDLR) . . . 265
15.12 Ratio de entrega de paquetes de datos (AntOR-RDLR) . . . 266
15.13 Throughput - caso a (AntOR-UDLR) . . . 267
15.14 Throughput - caso b (AntOR-UDLR) . . . 267
15.15 Ratio de entrega de paquetes de datos - caso a (AntOR-UDLR) . . . 268
15.16 Ratio de entrega de paquetes de datos - caso b (AntOR-UDLR) . . . 268
15.17 Retardo medio extremo a extremo - caso a (AntOR-UDLR) . . . 268
15.18 Retardo medio extremo a extremo - caso b (AntOR-UDLR) . . . 269
15.19 Sobrecarga en el número de paquetes (AntOR-UDLR) . . . 269
15.20 Sobrecarga en el número de bytes (AntOR-UDLR) . . . 270
15.21 Throughput (AntOR-v2) . . . 271

15.22 Ratio de entrega de paquetes de datos - caso a (AntOR-v2) 271
15.23 Ratio de entrega de paquetes de datos - caso b (AntOR-v2) 272
15.24 Retardo medio extremo a extremo (AntOR-v2) 272
15.25 Jitter - caso a (AntOR-v2) 273
15.26 Jttter - caso b (AntOR-v2) 273
15.27 Sobrecarga en el número de paquetes - caso a (AntOR-v2) 274
15.28 Sobrecarga en el número de paquetes - caso b (AntOR-v2) 274
15.29 Sobrecarga en el número de bytes (AntOR-v2) 275
15.30 Throughput - caso a (HACOR) 276
15.31 Throughput - caso b (HACOR) 276
15.32 Ratio de entrega de paquetes de datos - caso a (HACOR) 277
15.33 Ratio de entrega de paquetes de datos - caso b (HACOR) 277
15.34 Retardo medio extremo a extremo - caso a (HACOR) 278
15.35 Retardo medio extremo a extremo - caso b (HACOR) 278
15.36 Jitter - caso a (HACOR) 279
15.37 Jitter - caso b (HACOR) 279
15.38 Sobrecarga en el número de paquetes - caso a (HACOR) 280
15.39 Sobrecarga en el número de paquetes - caso b (HACOR) 280
15.40 Sobrecarga en el número de bytes - caso a (HACOR) 281
15.41 Sobrecarga en el número de bytes - caso b (HACOR) 281
15.42 Throughput (PAntOR) 282
15.43 Ratio de entrega de paquetes de datos - caso a (PAntOR) 282
15.44 Ratio de entrega de paquetes de datos - caso b (PAntOR) 283
15.45 Retardo medio extremo a extremo - caso a (PAntOR) 283
15.46 Retardo medio extremo a extremo - caso b (PAntOR) 284
15.47 Jitter - caso a (PAntOR) 284
15.48 Jitter - caso b (PAntOR) 285
15.49 Sobrecarga en el número de paquetes (PAntOR) 285
15.50 Ratio de entrega de paquetes de datos (PAntOR-MI) 286

List of Tables

2.1 Capacities of self-organization . . . 6
2.2 Applications of mobile ad hoc networks . . . 11

6.1 Routing table (AntOR) . . . 53
6.2 Routing structure of node B (AntOR) . . . 54
6.3 Neighbor table (AntOR) . . . 54
6.4 Neighbor structure of node A (AntOR) . . . 55
6.5 Routes for node A (AntOR) . . . 70
6.6 Routes for node D (AntOR) . . . 70
6.7 Routes for node B (AntOR) . . . 71
6.8 Routes for node C (AntOR) . . . 71
6.9 Routes for node A in diffusion process (AntOR) . . . 71
6.10 Routing table (AntOR-DLR) . . . 73

7.1 AntOR-DLR parameters . . . 93
7.2 AntOR-DNR parameters . . . 94
7.3 AntOR-RDLR parameters . . . 95
7.4 AntOR-UDLR parameters . . . 96
7.5 AntOR-v2 parameters . . . 97
7.6 HACOR parameters . . . 98
7.7 PAntOR parameters . . . 99
7.8 PAntOR-MI parameters . . . 100
7.9 Internal characteristics of AntOR-DLR . . . 101
7.10 Internal characteristics of AntOR-DNR . . . 101
7.11 Internal characteristics of AntOR-RDLR . . . 102
7.12 Internal characteristics of AntOR-UDLR . . . 102
7.13 Internal characteristics of AntOR-v2 . . . 103
7.14 Internal characteristics of HACOR . . . 103
7.15 Internal characteristics of PAntOR . . . 104
7.16 Internal characteristics of PAntOR-MI . . . 104

10.1 Capacidades de la auto-organización . . . 156
10.2 Aplicaciones de las redes móviles ad hoc . . . 161

14.1 Tabla de encaminamiento (AntOR) . . . 205
14.2 Estructura de encaminamiento del nodo B (AntOR) . . . 206
14.3 Tabla de vecinos (AntOR) . . . 207
14.4 Estructura de vecinos del nodo A (AntOR) . . . 207
14.5 Rutas para el nodo A (AntOR) . . . 223
14.6 Rutas para el nodo D (AntOR) . . . 223

14.7 Rutas para el nodo B (AntOR) . . . 223
14.8 Rutas para el nodo C (AntOR) . . . 224
14.9 Rutas para A en el proceso de difusión (AntOR) . . . 224
14.10 Tabla de encaminamiento (AntOR-DLR) . . . 226

15.1 Parámetros AntOR-DLR . . . 247
15.2 Parámetros AntOR-DNR . . . 248
15.3 Parámetros AntOR-RDLR . . . 249
15.4 Parámetros AntOR-UDLR . . . 250
15.5 Parámetros AntOR-v2 . . . 251
15.6 Parámetros HACOR . . . 252
15.7 Parámetros PAntOR . . . 253
15.8 Parámetros PAntOR-MI . . . 254
15.9 Características internas de AntOR-DLR . . . 255
15.10 Características internas de AntOR-DNR . . . 255
15.11 Características internas de AntOR-RDLR . . . 256
15.12 Características internas de AntOR-UDLR . . . 256
15.13 Características internas de AntOR-v2 . . . 257
15.14 Características internas de HACOR . . . 257
15.15 Características internas de PAntOR . . . 258
15.16 Características internas de PAntOR-MI . . . 258

List of Algorithms

6.1 Calculation of disjoint link routes (AntOR-DLR) 74
6.2 Calculation of disjoint node routes (AntOR-DNR) 76
6.3 Route calculation (AntOR-RDLR) 78
6.4 Neutralization of link failure (AntOR-UDLR) 80
6.5 Link failure management (HACOR) 84
6.6 Route setup (PAntOR-MI) 89

14.1 Cálculo de rutas de enlace disjunto (AntOR-DLR) 227
14.2 Cálculo de rutas de nodo disjunto (AntOR-DNR) 229
14.3 Cálculo de rutas (AntOR-RDLR) 231
14.4 Neutralización de fallo de enlace (AntOR-UDLR) 234
14.5 Gestión de fallos de enlace (HACOR) 237
14.6 Establecimiento de ruta (PAntOR-MI) 242

Acronyms

ACO	*Ant Colony Optimization.*
AI	*Artificial Intelligence.*
AntOR	*Ant Optimized Routing.*
AntOR-DLR	*AntOR Disjoint Link Route.*
AntOR-DNR	*AntOR Disjoint Node Route.*
AntOR-RDLR	*AntOR Restrictive Disjoint Link Route.*
AntOR-UDLR	*AntOR Unicast Disjoint Link Route.*
AODV	*Ad Hoc On-Demand Distance Vector.*
API	*Application Programming Interface.*
AS	*Ant System.*
BRP	*Bordercast Resolution Protocol.*
CBR	*Constant Bit Rate.*
COD-OLSR	*Coded-Optimized Link State Routing.*
DARPA	*Defense Advanced Research Projects Agency.*
DSDV	*Destination-Sequenced Distance-Vector.*
DSR	*Dynamic Source Routing.*
DSSS	*Direct Sequence Spread Spectrum.*
DYMO	*Dynamic Manet On-Demand.*
ERS	*Expanding Ring Search.*
FBCB2	*Force XXI Battle Command, Brigade-and-Below.*
FSR	*Fisheye State Routing.*

GloMo	*Global Mobile Information Systems.*
GSM	Global System for Mobile Comunications.
GSR	*Global State Routing.*
HACOR	*Hybrid ACO Routing.*
IANA	*Internet Assigned Numbers Authority.*
IARP	*Intra-Zone Routing Protocol.*
IEEE	*Institute of Electrical and Electronic Engineers.*
IERP	*Inter-Zone Routing Protocol.*
IETF	*Internet Engineering Task Force.*
INET	*Integrated Network Enhanced Telemetry.*
InterRT	*InterRT Interzone Routing Table.*
IntraRT	*IntraRT Intrazone Routing Table.*
ISM	*Industrial Scientific and Medical.*
LAR	*Location-Aided Routing.*
LMR	*Lightweight Mobile Routing.*
LPR	*Low-Cost Packet Radio.*
LR	*Local Retransmission.*
MANET	*Mobile Ad Hoc Network.*
MID	*Multiple Interface Declaration.*
MPI	*Message Passing Interface.*
MPR	*Multipoint Relay.*
NED	*Network Description.*
NS-2	*Network Simulator 2.*
NS-3	*Network Simulator 3.*
NTDR	*Near-Term Digital Radio.*
OFDM	*Orthogonal Frecuency-Division Multiplexing.*

OLSR	*Optimized Link State Routing.*
OMNeT	*Objective Modular Network Testbed.*
OSI	Open System Interconnection.
OSPF	*Open Shortest Path First.*
oTcl	*Object-Oriented Tool Language.*
PAN	*Personal Area Network.*
PAntOR	*Parallel AntOR.*
PAntOR-MI	*PAntOR Multiple Interface.*
PBA	*Proactive Backward Ant.*
PDA	*Personal Digital Assistant.*
PFA	*Proactive Forward Ant.*
PMP	*Proactive MANET Protocol.*
POSIX	*Portable Operating System Interface.*
PVM	*Parallel Virtual Machine.*
QoS	*Quality of Service.*
RBA	*Reactive Backward Ant.*
RFA	*Reactive Forward Ant.*
RFC	*Request For Comments.*
RMP	*Reactive MANET Protocol.*
ROAM	*Routing On-Demand Acyclic Multi-path.*
RRBA	*Route Repair Backward Ant.*
RREP	*Route Reply.*
RREQ	*Route Request.*
RRFA	*Route Repair Forward Ant.*
RWP	*Random Waypoint.*
SCS	*Shortest Common Supersequence.*
SURAN	*Survivable Radio Networks.*
TBRPF	*Topology Dissemination Based on Reverse-Path Forwarding.*

TC	*Topology Control*.
TORA	*Temporally-Ordered Routing Algorithm*.
TSP	*Traveling Salesman Problem*.
TTL	*Time To Live*.
UDP	*User Datagram Protocol*.
ULN	*Unicast Link Notification*.
VoIP	*Voice-Over Internet Protocol*.
WLAN	*Wireless Local Area Network*.
WRP	*Wireless Rou-ting Protocol*.
WSN	*Wireless Sensor Network*.
ZRP	*Zone Routing Protocol*.

Part I

Description of the Research

Chapter 1

Introduction

The term ad hoc is a Latin expression that literally means "for this". Generally it is referred to a solution, specifically made for a problem or precise purpose and, therefore, it cannot be generalized or used for alternative purposes. It is utilized to refer to something that is only suitable for a determined purpose. In the broad sense, ad hoc can translate as "specific".

The *Institute of Electrical and Electronic Engineers* (IEEE) [IEE99] defined the ad hoc networks as those networks composed only by stations, each one of them being within the range of coverage for any of the others through a wireless medium. An ad hoc network is typically created dynamically and its main singularity is its limitation both temporal and spatial. These restrictions allow the networks to create and dissolve in a sufficiently simple and practical manner. Formally, a wireless ad hoc network presents the following characteristics [Fee01]:

- **Wireless**: The nodes or stations communicate via non-guided transmission mediums (radio, infrared and so on).
- **Ad hoc**: The network is temporary and is set dynamically in an arbitrary manner by a set of nodes according to what is needed.
- **Autonomous and without infrastructure**: The network does not depend on an established infrastructure or any centralized management.
- **Multi-hop**: It does not need dedicated routers. Each node acts as a router and it forwards packets to other nodes to facilitate the exchange of information among the members of the network.

In addition, the nodes can be equipped with mobility. In this case, these networks are called *Mobile Ad Hoc Network* (MANET). The topology of this kind of network is dynamic because of the constant movement of the participant nodes, making communication patterns among members of the network to evolve continuously.

In definitive, the mobile ad hoc networks eliminate the imposed restrictions by fixed infrastructures, allowing the devices to create and join networks suddenly, making them suitable to adapt to virtually any application.

The rest of this chapter is organized as follows: section 1.1 presents the objective of this research work. Section 1.2 briefly comments on the problematic nature of routing in mobile ad hoc networks. The context in which the research has been developed is described in section 1.3 The contributors of this thesis are presented in section 1.4. Finally, section 1.5 summarizes the structure from memory.

1.1 Objetives

Mobile ad hoc networks have special characteristics that should be taken into account when implementing a routing protocol. There are many solutions (*Request For Comments* (RFC) 3626 [CJ03], RFC 3561 [PBRD03], 4728 [JHM07], 3684 [OTL04],...). All these protocols are valid solutions but they usually take determined characteristics of topology and some particular scenarios as designing premises, not being particularly suitable if there are drastic changes in the dynamic topology of the ad hoc network.

There is a group of algorithms or routing protocols so-called bioinspired having their adaptive capability as an essential feature, something especially noteworthy in this kind of environments. Within these algorithms there have been references in the literature especially based on the concept of collective intelligence, that is, those that apply the social behavior of insects and other animals to solve problems. The *Ant Colony Optimization* (ACO) algorithm constitutes the starting point of these algorithms. ACO algorithms are based on the collective behavior of the ants in their search for food and to bring it back to the nest.

AntHocNet [DC04, DCDG04, Duc07, WDR08, KO08, DCDG08] is a benchmark in the area of ACO routing protocols for mobile ad hoc networks. Its adaptive nature means its performance metrics are the best overall. Nevertheless, it presents some scalability issues in highly dynamic scenarios. This work seeks to correct such deficiency.

1.2 Problem Identification

The objective of a routing protocol for mobile networks is to get the sending of a message from one node to another without the existence of a direct link. The most routing protocols for mobile ad hoc networks come from adaptations carried out on protocols of fixed networks, with the main problem being the amount of failures that occur in communication because of the mobility of the nodes. It is therefore essential that the design of specific algorithms adapt quickly to the peculiarities of this type of network.

1.3 Research Context

This Doctorate Thesis has been made within the research group GASS (analysis group, security and systems, group 910623 of the catalogue of groups recognized by the UCM) as part of the activities of various research projects totaling more than 5 years of work.

This research begins in the context of a research project of the Programa de Fomento de la Investigación Técnica (PROFIT) of Ministerio de Industria, Turismo y Comercio (MITyC) with the help of Safelayer Secure Communications, S. A., Spanish company which constitutes a reference in the area of Tecnologías de la Información y las Comunicaciones (ICT area) both nationally and internationally. More specifically, with the work developed in the project Semantic & Ambient Trust Technologies (reference FIT-360000-2007-48).

The Programa de Fomento de la Investigación Técnica is an instrument through which the Government articulates a set of calls for public help, intended to stimulate the companies and other entities to carry out activities of research and technological development. According to the established objectives set out in the Plan National de la Investigación Científica, Desarrollo e Innovación Tecnológica (I+D+i), in the part dedicated to the Fomento de la Investigación Técnica.

Research continues in the projects: Semantic & Ambient Trust Technologies II (Subprogram Avanza I+D, reference TSI-020100-2008-365), Semantic & Ambient Trust Tech-

nologies III (Subprogram Avanza I+D, reference TSI-020100-2009-374), and Trust as a Service (Subprogram Avanza Competitividad I+D+I, reference TSI-020100-2010-482), and ends with the project Privacy-aware Accountability for Trustworthy Future Internet (Subprogram Avanza Competitividad I+D+I, reference TSI-020100-2011-165).

The Subprogram Avanza of MITyC, in its various modalities, comes to being the continuation of the Program PROFIT. In all the projects AVANZA mentioned above the company Safelayer Secure Communications, S. A also participates.

1.4 Summary of Contributions

From the point of view of the ACO routing in mobile ad hoc networks, the results of this Thesis comprises various protocols (see Figure 1.1).

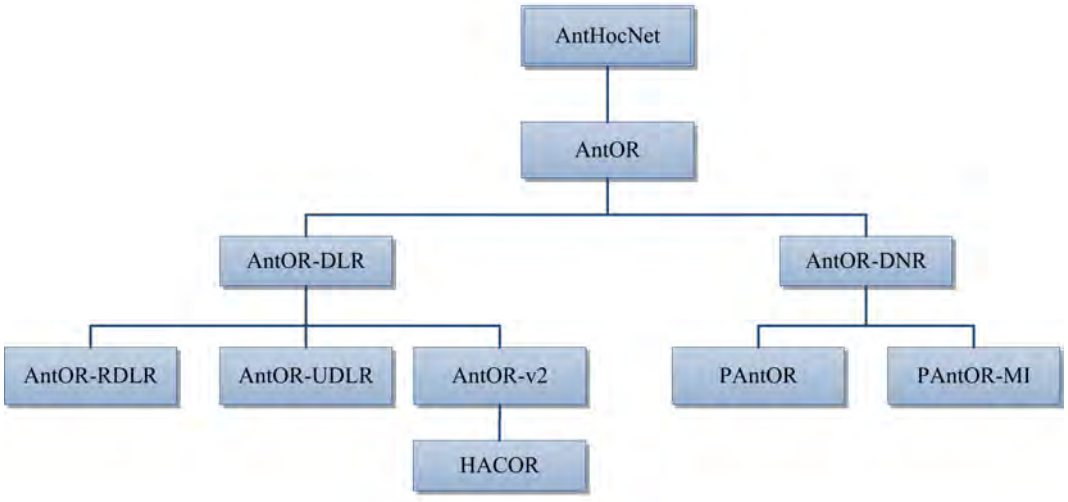


Figure 1.1: Evolution scheme of AntOR

All of them derive from a base protocol so-called *Ant Optimized Routing* (AntOR) [GVRCSO10]. Like its predecessor AntHocNet, AntOR is hybrid in the sense that it contains routing both reactive and proactive elements, combining a reactive process of route setup with a proactive process of maintenance and exploration of new routes.

AntOR presents two variants: AntOR Disjoint Link Routing *AntOR Disjoint Link Route* (AntOR-DLR) [RCSOGVK11b] where the routes do not share links and *ANTOR DISJOINT NODE ROUTE* (ANTOR-DNR) [RCSOGVK11b] where the routes do not share nodes.

Both variants originate a set of protocols that are successive refinements from the original protocols.

The disjoint - link version originates a set of sequential protocols: *AntOR Restrictive Disjoint Link Route* (AntOR-RDLR) [GVRCSOK12b], *AntOR Unicast Disjoint Link Route* (AntOR-UDLR) [RCSOGV12a, RCSOGV12b], AntOR-v2 [RCGV12, RCGVSOK13, RCSOGV13b] and *Hybrid ACO Routing* (HACOR) [RCSOGV13a, RCSOGVH13]. AntOR-RDLR allows the generation of more alternative routes. AntOR-UDLR reduces the number of control messages on link failures. AntOR-v2 and HACOR are the two more evolved variants, providing new techniques such as storage of control packets and outdated routes

management, as well as improvements in the link failures management and route exploration.

The disjoint - node version originates a set of parallel protocols: *Parallel AntOR* (PAntOR) [RCSOGVK11b, GVRCSO13] and *PAntOR Multiple Interface* (PAntOR-MI) [GVRCSOK12a]. PAntOR is a large-grained parallelization of AntOR making use of multiprocessor programming architectures based on a shared memory system through the Posix Thread standard. PAntOR-MI is a multi-interface variant of PAntOR.

1.5 Outline of the Thesis

This thesis is structured as follows:

Chapter 2 carries out a state of the art in mobile ad hoc networks including a chronological review of the evolution in mobile ad hoc networks, an analysis of the basic characteristics of this type of network, a presentation of the communication protocol that is currently utilized in them, a classification of mobile ad hoc networks and a summary of the main applications of mobile ad hoc networks.

Chapter 3 addresses the problem of the routing in mobile ad hoc networks. It begins by pointing out the non-applicability of the standard solutions and the need for robust and adaptive protocols. Subsequently, it presents a classification of routing protocols for mobile ad hoc network, describing in detail two of singular importance: AODV and OLSR.

Chapter 4 focuses on the Ant Colony Optimization, area of *Artificial Intelligence* (AI) which is inspired by the behavior of ants in nature and which is essential for understanding the functioning of the routing protocols for mobile ad hoc networks developed in this Thesis.

Chapter 5 analyses one of the multiple applications of the Meta-heuristic ACO: so-called ACO routing which constitutes all a model in the design of routing protocols for mobile ad hoc networks. This chapter reviews the State of the art of the protocols in detail ACO routing for mobile ad hoc, doing emphasis in AntHocNet, ACO routing hybrid protocol which is, without a doubt, a reference in the area.

Chapter 6 contains the contributions of this work: a family of ACO routing protocols for mobile ad hoc networks constructed from a base protocol so-called AntOR and inspired by AntHocNet, which inherits its hybrid character.

Chapter 7 contains the simulation results carried out in *Network Simulator 3* (NS-3) [NS3] software.

Finally, Chapter 8 shows the main conclusions extracted from this work as well as some future lines of research.

Chapter 2

Mobile Ad Hoc Networks

The overall objective of this chapter is to facilitate the understanding of what mobile ad hoc networks are. First, a chronological review of the evolution of mobile ad hoc networks is carried out. Subsequently, the basic characteristics of this type of network are analyzed. Then attention is paid to the communication protocol that is currently used in such networks, the IEEE 802.11. Following this, a classification of mobile ad hoc networks is shown. Then, the main applications of mobile networks ad hoc are presented. The chapter ends with a brief summary of what has been exposed.

2.1 Evolution Historical

In a few years, the field of mobile ad hoc networks has experienced a rapid visible expansion of the proliferation of low-cost wireless devices such as laptops, personal digital assistants *Personal Digital Assistants* (PDAs), mobile phones, etc.

At the beginning of the 70s a pioneer radio worker from the University of Hawaii presented the first system that uses the medium of radio to transmit information. Widely known as ALOHA [Abr70], was developed by Abramson and Kuo.

The work done in Hawaii took place in 1972 to the development of a distributed architecture consisting of a radio broadcast network with central minimal control called PARNET sponsored by *Defense Advanced Research Projects Agency* (DARPA). The project helped to establish the concept of mobile ad hoc networks. PARNET allowed direct communication amongst mobile users over large geographical areas, shared bandwidth and protection against the effects of multiple paths.

The rapid advances of the technology of radio in the 70s resulted in the emergence of multiple systems of mobile communication like cellular and wireless telephones, radio search systems, mobile satellites, etc.

Subsequently, DARPA developed the project *Survivable Radio Networks* (SURAN) in 1983 which covers the tasks of scalability of the network, security, process ability and power management. Efforts were devoted to developing low cost devices and with little expenditure of energy that could withstand the advanced routing protocols, scale the networks to thousands of nodes and support for attacks on security. The result was the emergence of the technology known as *Low-Cost Packet Radio* (LPR) in 1987.

Midway through the 90's produced a new advance with the arrival of radio cards 802.11 for personal computers and laptops. [FL01, Jai03]] propose the idea of a collection of mobile hosts with a minimal infrastructure for the first time, and the IEEE fixes the term *ad hoc networks*.

During the same time, Department of Defense from the United States continued working with projects such as *Global Mobile Information Systems* (GloMo) or the *Near-Term Digital Radio* (NTDR). The goal of GloMo was to allow the multimedia connectivity of type Ethernet, at any time and in any place, among wireless devices. NTDR are protocols that are based on two components: clustering and routing. Clustering algorithms dynamically organize a network in group leaders and group members. Leaders form the backbone of the network and members communicate with each other through this column. NTDR was initially a prototype for Navy from the United States and currently some countries use it as a base for other protocols.

The definition of standards as IEEE 802.11 [IEE99] caused the rapid growth of mobile networks not only in military fields, but also in the commercial world.

2.2 Characteristics

As its own name indicates the main characteristic of a mobile ad hoc network is the mobility of the nodes that can change position quickly. The need to create networks quickly in places without infrastructure often involves nodes exploring the area, and in some cases, they may join to achieve a goal. The type of mobility which the nodes develop may have an influence at the time of choosing the routing protocol that increases the performance of the network.

Another important aspect in mobile ad hoc networks is so-called self-organization, which is studied in depth in [Fee01]. The main idea is based on the coordination and collaboration of all the nodes in the network to achieve the same objective. Several methods of self-organization network in general and for ad hoc networks in particular are proposed. Self-organization can be broken down into capacities shown in 2.1.

Table 2.1: Capacities of self-organization

Capacity	**Description**
Self-reparation	Mechanisms allow the failure to detect, locate and repair faults automatically being able to distinguish the cause of the error. For example, overhead or malfunction.
Self-configuration	Methods of generation suitable in function of the current situation depending on environmental circumstances. For example, connectivity or parameters of quality of service.
Self-management	The ability to maintain devices or networks depending on the current parameters of the system.
Adaptation	Adequacy to changes in environmental conditions. For example, change in the number of neighbor nodes.

Then the rest of characteristics of mobile ad hoc networks are presented:

- **Absence of infrastructure**: On the contrary to conventional networks that have the existence of physical elements, mobile networks are formed autonomously.

- **Dynamic topology**: Nodes can move arbitrarily destroying some links whilst others are created when a node is approaching others which previously had out of its reach.
- **Limited Bandwidth**: In most occasions, it will be lower than the one of a wired connection, moreover affected by the interference of electromagnetic signals.
- **Variation in the capacity of links and nodes**: Nodes can have several interfaces of radio which differ in transmission/reception capacity and the frequency band in which they work. This characteristic complicates the development of routing protocols to a large degree.
- **Energy conservation**: Some or all nodes from a mobile ad hoc network are fed by batteries and they do not have the possibility of recharging them. For these nodes, the most important criterion at the time of designing systems and protocols will be the optimization of energy conservation.
- **Scalability**: In many applications ad hoc networks may have thousands of nodes which lead to difficulty in tasks such as addressing, routing, location management, configuration management, interoperability, security and so on.
- **Lack of security**: The security plays an important role in the ad hoc networks given the vulnerable nature of wireless links that form. The protocols of routing must provide a secure communication. The routing protocols should supply a secure communication. There are research areas in this sense suggesting to include data from external sensors and geographical and topographical information in the own routing algorithm.
- **Multi-hop routing**: Nodes act as routers to retransmit the exchanged packets between nodes of which reach does not allow direct communication.
- **Unpredictable environment**: Mobile ad hoc networks may occur on lands in which situations are not the most optimal due to dangerous and unknown conditions. There may be cases where the nodes are destroyed, damaged or begun to get failures.
- **Behavior of the terminals**: one of the main keys for a mobile ad hoc network has a suitable functioning is the confidence that each node must have over others. Without this confidence, it would be impossible to create a routing protocol as information should be transmitted by several intermediate nodes. Normally, the routing protocols that discover intermediate terminals are based on responses that the nodes give on the cost of communication. There are malicious nodes which could intentionally report on the costs in an incorrect way with the aim of receiving all packets, to be able to manipulate them, to alter them, or even to delete them. Some solutions to this matter are found in [PHM+06].

Figure 2.1 presents a typical example of mobile ad hoc network.

2.3 Standard IEEE 802.11

IEEE 802.11 is a standard of communication protocol that defines the use of the two lower levels of the Open System Interconnection (OSI) architecture (physical layer and data link layer), specifying its rules of operation in a wireless network. The first proposal of this standard maintained transmission rates of 1 and 2 Mbps in the frequency band *Industrial*

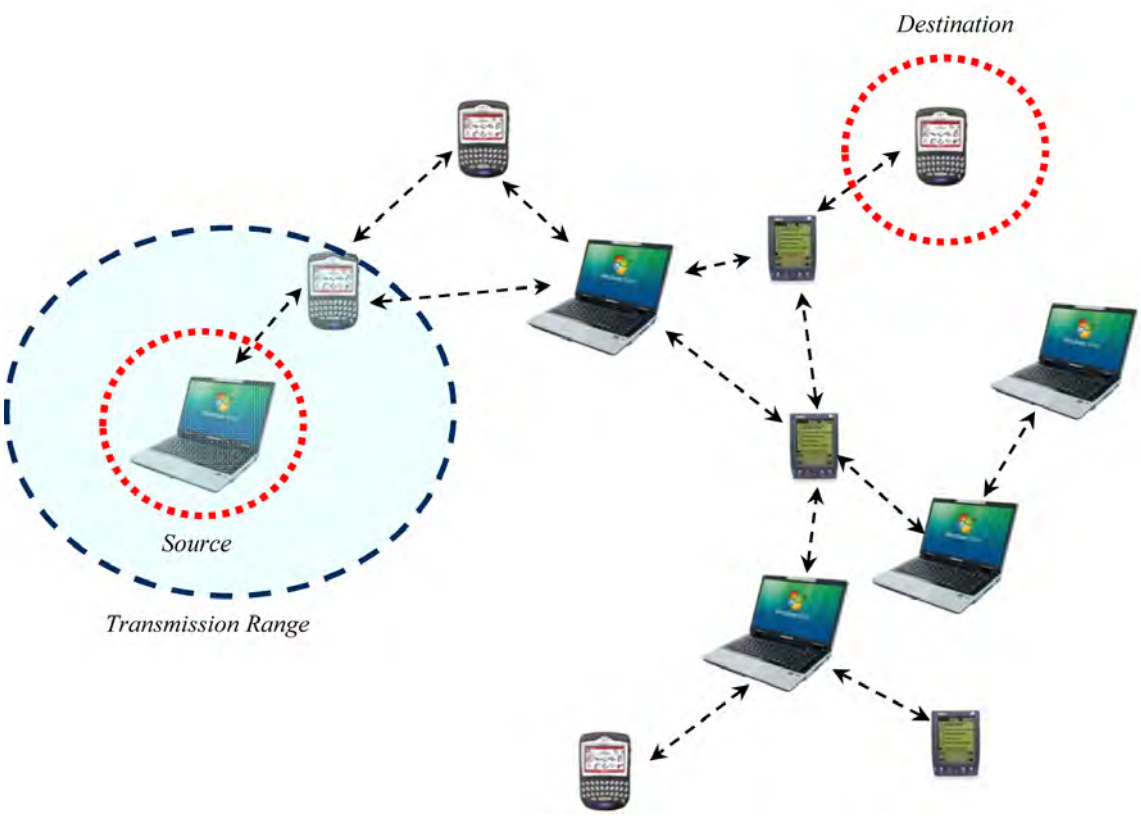


Figura 2.1: Mobile ad hoc network

Scientific and Medical (ISM), located at 2.4 GHz. In addition, they were specified infrared and radio channel as technologies in the physical layer. Over the years it has been come to different versions of the standard. The most important are listed below:

- **IEEE 802.11a**: Until 54 Mbps to 5 GHz. It utilizes the technology *Orthogonal Frecuency-Division Multiplexing* (OFDM) in the physical layer.
- **IEEE 802.11b**: Until 11 Mbps to 2.4 GHz. Currently, it is the most used. It utilised the technology *Direct Sequence Spread Spectrum* (DSSS) in the physical layer.
- **IEEE 802.11e**: It aims to provide *Quality of Service* (QoS) for use in services as *Voice-Over Internet Protocol* (VoIP) and Streaming. An approach to provide quality of service is the one of differentiating packets by classifying them into a small number of types of services and using priority mechanisms to provide a quality service suitable to each traffic.
- **IEEE 802.11f**: : It develops specifications for the implementation of points of access and distribution systems to prevent problems of interoperability between different manufacturers and distributors of equipment.
- **IEEE 802.11g**: Until 54 Mbps to 2.4 GHz. It supports both OFDM and DSSS in the physical layer.

2.4 Classification

The terminology of ad hoc networks is not yet very suited and there isn't a clear classification. Several classifications are then exposed, which locate the place where mobile ad hoc networks are.

There are ad hoc networks *with infrastructure* where the nodes move while they communicate with a fixed base station. When a node moves outside the range of a fixed station enters the reach of another station. When a node moves outside the range of a fixed station, it enters the reach of another station. On the other hand there are networks ad hoc *without infrastructure* where there are no fixed base stations and all the nodes in the network need to act as routers. **Mobile ad hoc networks are ad hoc networks without infrastructure**.

Another classification of mobile ad hoc networks includes *networks of a single-hop* and *multi-hop networks*. Nodes in the networks in a single hop only communicate with the nodes that it has at its reach. On multi-hop ad hoc networks, nodes that cannot communicate directly use intermediate nodes to retransmit the information. **Mobile ad hoc networks are multi-hop ad hoc networks**.

Finally there is a classification that includes mobile ad hoc networks as an independent type. Three types of ad hoc networks are included:

- **Mobile ad hoc networks**.
- **Sensor networks**: Also known as *Wireless Sensor Network* (WSN). Formed of sensory devices, usually consisting of a traditional sensor and an analogical-digital converter. The process unit is composed of a microprocessor and a small memory. They can include location and mobility systems. In these networks the number of nodes is usually much higher than in a mobile ad hoc network, but mobility is considered to be little or null (only they change the topology with the loss or disconnection of nodes). The flow of information is common from many sources to a node called *sink* that is responsible for processing the information and sending it to the destination.
- **Hybrid networks**: Also called mixed, they are ad hoc networks that use IP infrastructure if they are available.

At the same time mobile ad hoc networks can be divided into two types depending on whether they are connected or not to other networks:

- **Autonomous mobile ad hoc networks**: They are networks that are not connected to any other network. The nodes of the network are uniquely identified through an IP address with the unique premise that it is different from any other node in the network.
- **Subordinate mobile ad hoc networks**: They are networks connected to one or more external networks. You are forced to use a topological IP addressing correct and routable globally. A typical example of subordinate mobile ad hoc network is a mobile ad hoc network which is part from Internet.

2.5 Applications

It is easy to find situations where you can see the usefulness of mobile ad hoc networks. One of the most classic examples (although also discussed) is a working meeting: a group

of people with laptops or PDAs. They are different companies and their addresses are therefore different. Perhaps the room has Internet access and they may use for example mobile IP, but what stroll its datagram across town or across the country when they are in the same room? Their machines are likely to be equipped with infrared or Bluetooth ports that allow them to form a network for the occasion. In some cases, there will simply not be support infrastructures. We think in isolated populations or difficult terrain, situations of emergency, natural disaster where infrastructures have disappeared and so on.

Another example is the so-called *Personal Area Network* (PAN),networks formed by the devices of a person, such as his watch, his agenda and his mobile phone. A network may therefore want to come into contact with the network of another person who at that moment is next.

The ability to deploy them immediately and the non-dependence on a single point of failure make these networks very interesting for military use. The military field is possibly the most developed currently. Thus, the U.S. Army already has a system based on this kind of networks, the *Force XXI Battle Command, Brigade-and-Below* (FBCB2) .One of its objectives is to distinguish the own forces of the enemy forces, offering the soldiers a view of the battlefield similar to a videogame. The teams of the generation immediately before were based on communications by satellite, with latencies of five minutes. In April of 2003 the FBCB2 was used in WWII of the Gulf, which probably supposed the first use under real fire from a mobile ad hoc network.

Another reason whereby a mobile ad hoc network may be advantageous is the cost. Although there is a network infrastructure, if it belongs to a foreign entity, it is likely that it charges us for its use, while if we have our machines deployed we will already have a network without additional cost. For example, the cars that pass by a freeway could easily form a mobile ad hoc network, independent of its capacity of connecting to other networks as Global System for Mobile Comunications (GSM) or similar. Finally, we suppose we have stations able to communicate using a satellite. These communications equipment are expensive, but it is enough to have some capacity to connect the satellite so that all connectivity is disposed of. Also, not all of those able to connect to the satellite would be connected simultaneously.

The most notable quality of mobile ad hoc networks is its flexibility. The fact that may be established anywhere and at any time without infrastructure, administration and pre-configuration, make them very attractive for a wide range of fields of application.

Table 2.2 shows a classification of present and future applications of ad hoc networks, as well as services offered [BR04].

Table 2.2: Applications of mobile ad hoc networks

Tactical networks	Communications on military operations. Automated Battlefields.
Sensor networks	Data collection in real time, usually highly correlated in space and time.
Rescue services and emergency	Search and rescue operations. Replacement of networks with infrastructure in the case of natural disasters.
Commertial environments	E-Commerce. Remote access to records from the clients from a centralized database. Mobile office. Vehicle services.
Personal networks and company	*Wireless Local Area Network* (WLAN) for homes or offices. PAN.
Educational applications	Configuration of ad hoc communications in meetings, conferences and congresses. Configuration of virtual classes.
Leisure	Multi-user videogames. Robot pets. Outside Internet access.
Localization services	Tracing services. Information services.

2.6 Summary

The objective of this chapter has been to present mobile ad hoc networks. We started with a brief review of their evolution. Subsequently, we have analyzed its main features highlighting the absence of infrastructure, dynamic topology and self-organization capacity. We have also commented on different versions of the Standard IEEE 802.11 or communication protocols of this type of networks. Finally, we have concluded with a classification of mobile ad hoc networks, as well as noting how its flexibility makes them suitable for a large number of applications.

Chapter 3

Routing in Mobile Ad Hoc Networks

This chapter is dedicated to the problematic nature of routing in mobile ad hoc networks, a fundamental aspect of this type of networks. It starts by looking at how the design should be for specific routing protocols for mobile ad hoc networks because of the nature of them, as well as characteristics or requirements that must be carried out to work properly, also commenting on the impossibility of using traditional solutions. Subsequently, different classifications of routing protocols for mobile ad hoc networks are shown: according to the state of information that stores the nodes of the network, depending on the structure and the procedure adopted for the discovery of the way to establish it. Then, special attention is paid to *Ad Hoc On-Demand Distance Vector* (AODV) and *Optimized Link State Routing* (OLSR) protocols, a reference in the area, which are to be used in the analysis of the performance of routing protocols developed in the present work. The chapter ends with a brief summary of the above in it.

3.1 Routing Protocols

In mobile ad hoc networks, the conventional protocols will either have a very poor performance, or will simply be inapplicable. As an alternative specific routing protocols are developed. They are often called level 2.5, since they are generally found above link protocols like IEEE 802.11 and below the network IP protocol.

The concept of routing basically comprises two activities. Firstly, it determines the optimal paths and, secondly, it transfers groups of information packets through the network. The algorithms use several metrics to calculate the best path in order for packets to arrive at their destination. These metrics are standard measures as the number of hops that are used by the algorithm to determine the optimal path. The process to determine the path initializes and maintains routing tables that contain the full information for each route. The information that is stored for each route varies from an algorithm to another.

Mobile ad hoc networks are built dynamically when a set of nodes created routes between themselves to get the connectivity between them. The mobile ad hoc network nodes can act as a source or destination for communication, but also as routers when a relationship between nodes cannot be carried out directly due to distance issues. In this way, multi-hop communications are created. A routing protocol in a mobile ad hoc network must provide a mechanism that maintains the routes towards the destinations given the movement of the nodes that may cause the destruction of the routes, and that it is necessary to find an alternative route in order to keep the communication between

the nodes.

The objective of a routing protocol for mobile networks is to get the sending of a message from a node to another without the existence of a direct link. The majority of routing protocols for mobile ad hoc networks come from adaptations made about protocols of fixed networks, the main problem being the amount of failures that occur in the communication due to the mobility of the nodes.

Routing protocols for mobile ad hoc networks should basically satisfy the following criteria [BR04]:

- *Minimal signaling*: The reduction of control messages helps to preserve the capacity of the batteries and the communication of the nodes.
- *Topology dynamic maintenance*: The algorithm should be able to quickly locate a new route when a link is broken.
- *Loop-Free*: It intends to avoid the problem of having packets circulating lost by the network.
- *Multi-hop capacity*: It must ensure that the forwarding of packets through the nodes of the network because of usually the destination is not within the reach of the source.
- *Minimum processing time*: Algorithms are required with computational calculus that are not excessively complex to reduce processing time and in this way, extend the life time of the battery.

In addition, it must support different modes of operation [MC04]:

- *Distributed*: Essential property of MANETs.
- *Inactive*: Routing protocols will have to be prepared for those periods of time in which nodes break their activity and remain inactive to save power.
- *On demand*: The adaptation of routing to particular traffic patterns of each situation makes it possible to reduce the cost of bandwidth and energy, but the time of obtaining the route is extended.
- *Unidirectional links support*: Routing protocols Routing protocols have often been designed and work correctly only with bidirectional links and this should not be so, because in practice we can meet with the existence of unidirectional links to be key for the information exchange of in mobile ad hoc networks.

Many routing protocols for MANET based on these criteria are designed.

The finality of *Internet Engineering Task Force* (IETF) MANET WG [MAN] is to standardize the functionality of an IP routing protocol for applications of wireless routing within both static and dynamic topologies as consequence of the mobility of the nodes or other factors. Approaches are intended to be relatively general as they must be adequate in wireless and hardware environment multiple and targeted to scenarios where mobile ad hoc networks are deployed at the border of an IP infrastructure. Hybrid *mesh* infrastructure (for example, a mixture of fixed and mobile routers) should be supported by the specifications of the mobile ad hoc networks.

3.2 Classification of Routing Protocols

Since it was begun studying the mobile ad hoc networks they have been proposed different classifications of the Since mobile ad hoc networks have been studied, different classifications of the routing protocols have been proposed and are summarized in [JG07].

According to the state information that the nodes store of the network protocols can be classified into protocols based on topology and protocols based on the destination. In the first ones, each node makes decisions based on complete information of the topology of the network. The second ones are protocols that handle vectors of distances, in which each node exchanges the distances which other nodes know with neighbours.

Another classification proposes to divide protocols depending on the structure, differentiating several levels. Figure 3.1 presents this formulated classification, specifying some of the protocols of each category:

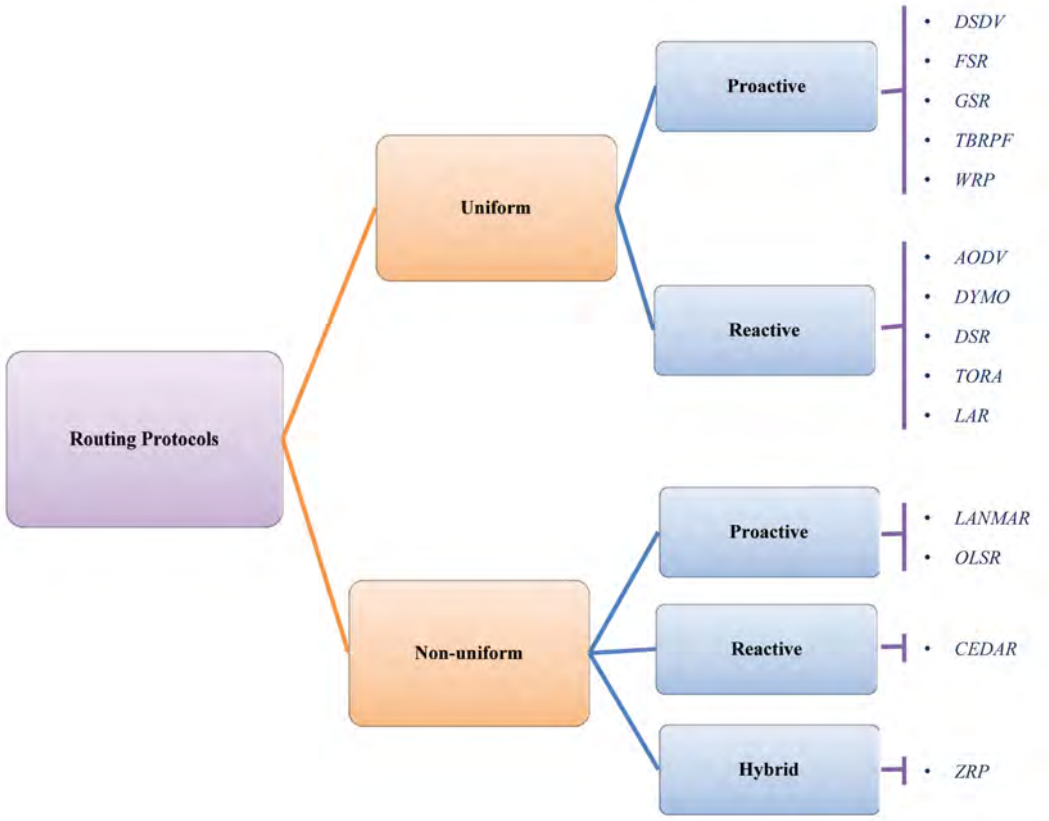


Figure 3.1: Taxonomy of routing protocols in mobile ad hoc networks

The first level refers to the homogeneity or heterogeneity of the functions of the nodes in the routing, distinguishing two types:

- *Uniform or flat structure protocols*: No node from the network performs a different role distinct to others; all of them send and reply to control messages in the same way.
- *Non-uniform protocols*: Typical of hierarchical structures in which some nodes develop specialty roles and can even provide special abilities in terms of computation,

energy or storage among others. This allows them to bear more complex algorithms, to reduce the overhead due to communication and to offer the possibility of load balancing while they maintain its characteristics, even with increases in the number of nodes in the network. On the contrary, they generate some cost of maintenance of the structure and often need the availability of heterogeneous nodes.

In these two previous categories, protocols present a new peculiarity related to the procedure adopted for the discovery of the path to establish and maintain. Undoubtedly this classification is the most widespread emerging for the following types of protocols:

- *Proactive protocols*: In this type of routing, each node maintains information on how to get to any other node in the network and exchanges this information with all of its neighbors. Routing information is normally stored in a number of different tables. The tables are updated periodically if the network topology changes. The difference among the protocols of this type is in the way of updating and detecting the routing information and the type of information that is stored in each table. The advantage that these protocols have is low latency since the routes are always available. However, this implies very high energy consumption in the nodes and it may produce an overhead of messages in the network due to the periodic flooding of messages. Then, the more representative proactive routing protocols are enumerated: *Destination-Sequenced Distance-Vector* (DSDV) [PB94], *Wireless Routing Protocol* (WRP) [MGLA95], *Global State Routing* (GSR) [CG98], *Fisheye State Routing* (FSR) [GHP02], OLSR [CJ03], *Topology Dissemination Based on Reverse-Path Forwarding* (TBRPF) [OTL04]. In general, these protocols treat to avoid loops in the routes, memory excessive consumption and reducing the size of the packets containing the information of the routing tables. Within the proactive protocols two subtypes of protocols can be distinguished according to their behavior: *event-driven*, which send packets with information about routes only when these have some change, and those that refresh the information periodically (*regular updated*). The OLSR protocol, which has been used for this work and that will be analyzed in detail in the next section, is within the second category.

- *Reactive protocols*: These protocols try to reduce the overhead produced by proactive protocols. To do this, they propose that when the nodes of the mobile ad hoc network do not have a route to a destination, they only calculate it when necessary, i.e. when the node has to begin an exchange of packets with the destination. The discovery of a route is normally done by a flood of request messages through the network. These protocols have a high latency, provoked by the discovery of routes. However, the overhead of messages across the network is reduced. Then the most representative reactive routing protocols are enumerated: AODV [PBRD03], *Dynamic Manet On-Demand* (DYMO) [CP09], *Dynamic Source Routing* (DSR) [JHM07], *Routing On-Demand Acyclic Multi-path* (ROAM) [RGLA99], *Lightweight Mobile Routing* (LMR) [CE95], *Location-Aided Routing* (LAR) [KV00], *Temporally-Ordered Routing Algorithm* (TORA) [PC01]. Most of them have the same cost of routing in the worst possible scenario since almost all follow the same philosophy for the discovery of routes.

- *Hybrid protocols*: By combining proactive and reactive protocols, hybrid protocols are formed which are intended to minimize the disadvantages of both. The idea of these protocols is that the nodes of the network work in a proactive manner with closest nodes and in a reactive manner with the rest of the nodes. The reactive part

controls the overhead and memory consumption by calculating routes only when they are needed. In contrast, the proactive part needs to periodically update the information stored and it maintains routes that perhaps will be never used, adding an unnecessary overhead. The best-known case of hybrid protocol is *Zone Routing Protocol* (ZRP) [HPS02].

The working group MANET has planned to develop two specifications standard routing protocols, so-called *Reactive MANET Protocol* (RMP) and *Proactive MANET Protocol* (PMP), although it is possible to choose a mixed approach. It will support IPv4 and IPv6, security requirements and other aspects, and will pay special attention to the OSPF-MANET protocol which is developed by the *Open Shortest Path First* (OSPF) WG [OSP]. OSFP WG develops extensions of OSPF protocol for different scenarios, as the OSPF-MANET extension of OSPF to mobile networks ad hoc.

The following sections describe a representative example of proactive, reactive, and hybrid protocol, more specifically, it explains the operation of the OLSR proactive protocol, the AODV reactive protocol and the ZRP hybrid Protocol. Emphasis is made on the first two, since literature refers to these to make a comparison between the performance of the new routing protocols.

3.3 OLSR Protocol

OLSR protocol belongs to the group of protocols for MANETs that are defined as RFC. OLSR is specified in RFC 3626 [CJ03], as an optimization of the classic protocol of link state or OSPF [Moy98], but adapted to mobile ad hoc networks.

As there are constant information updates about the proactive protocol and the routes to the nodes, it makes it available where needed. As indicated above, OLSR is *regular updated*, that is, every so often, information packets about routes are transmitted, although no changes have been detected.

The main contribution of OLSR that distinguishes it from other similar protocols are the optimizations that are performed in order for the produced overhead by periodic updates to be minimal. The way that information is collected about routes to whole network is through controlled flood: assigning certain nodes the responsibility of sending each other information, making it then reachable for the rest of the network, and checking duplicate data is not sent.

Due to optimizations that apply, it gets good results in large and dense networks. Their optimizations are noticeable in the performance of how much larger networks are, especially if the traffic is sporadic between pairs of nodes that vary irregularly, instead of regular communications between specific nodes.

3.3.1 Protocol Functioning

The first thing that a node does at the beginning is detect which other nodes it has connection with at link level. To this end, *Hello* messages are sent periodically. These messages are not retransmitted by the nodes receiving them, since their purpose is that nodes are taken to inform its neighbors of one-hop, i.e., nodes with existing connectivity at the link level. Another feature of the *Hello* message, apart from informing the node itself, is to announce the neighbor from the sender node. Thus, a node that listens to these messages not only discovers their neighbors to one-hop, but it also acquires knowledge of their neighbors in two-hop distance.

The key of the protocol is in the *Multipoint Relay* (MPR). Once a node knows the set of two-hop neighbors, choose from their neighbors in a hop a group of MPR nodes that relay their messages, in such a way that they provide access to all two-hop neighbors; in this way they will open the route toward any node in the network. The selected nodes as MPR are notified of their condition, maintaining information about who have been chosen as MPR (so-called MPR selectors) in a structure called a MPR selector set.

One of the tasks of the MPR is to retransmit broadcast messages generated by any of the nodes in its MPR selector set. In this way, messages reach the whole network, but trying to be the saturation minimum.

Another task that a selected node must carry out as MPR is to generate and to relay *Topology Control* (TC) messages, which are made known to the rest of the network nodes of which the sender is aware. TC messages are generated in a periodic way, and contain a list with the addresses of the nodes of the MPR selector set, nodes that have chosen to the sender of the message as MPR (unlike other protocols that would announce to any nearby node). With this, the information provided about the generated topology is minimal and that its dissemination by the network is done in a controlled way.

In other words, the MPR also feature to announce information about the topology of the network using controlled flooding, so that all participants know the route to the rest of the network. As indicated previously, the flood is performed by TC messages, generated by nodes which have been selected as MPR, and that are forwarded between the MPR in an efficient way rather than doing by mass broadcast.

It is also important to know the internal data structures that handles OLSR. In particular, the most relevant is the routing table. The information that we have of each node of the network in the routing table is an entry with the following fields:

R_dest_addr ***R_next_addr*** ***R_dist*** ***R_iface_addr***

That entry means that the node identified by the address *R_dest_addr* is at an estimated distance of *R_dist* hops, the neighbor with the *R_next_addr* interface address is the next hop on the route to *R_dest_addr*, and this neighbor is reachable through the local interface with the address *R_iface_addr*.

In OLSR the possibility is taken into account that a node has more than one network interface participating at the same moment in the network. Each interface address is associated with a main address, unique for each node. This main address will be the same as the address of the interface in the case that node has a unique interface using OLSR.

3.3.2 OLSR Packet Format

OLSR communicates are carried out using a common packet format for all data related to the protocol. In this way, it facilitates extensions of the protocol without breaking the compatibility of previous versions. Moreover this also facilitates to group different types of information into a same transmission.

The packets are embedded in *User Datagram Protocol* (UDP) datagrams for transmission over the network.

Each packet encapsulates one or several messages at the same time. The messages share a common header format, which allows a node to be able to accept and relay (if applicable) messages of an unknown type.

Messages can be transmitted by flooding to the network in its totality, or the transmission can be limited to nodes within a certain diameter -referring to number of hops- from the sender of the message. Thus transmitting a message to the neighborhood of a node is

just a special case of transmission by flooding. When control messages are transmitted, duplicate retransmissions are eliminated in a local way, since each node keeps information about control messages that has already transmitted previously.

Packets in OLSR use using UDP 689 port, assigned by *Internet Assigned Numbers Authority* (IANA).

The fields of any OLSR packet (omitting IP and UDP headers) are indicated in the Figure 3.2 .

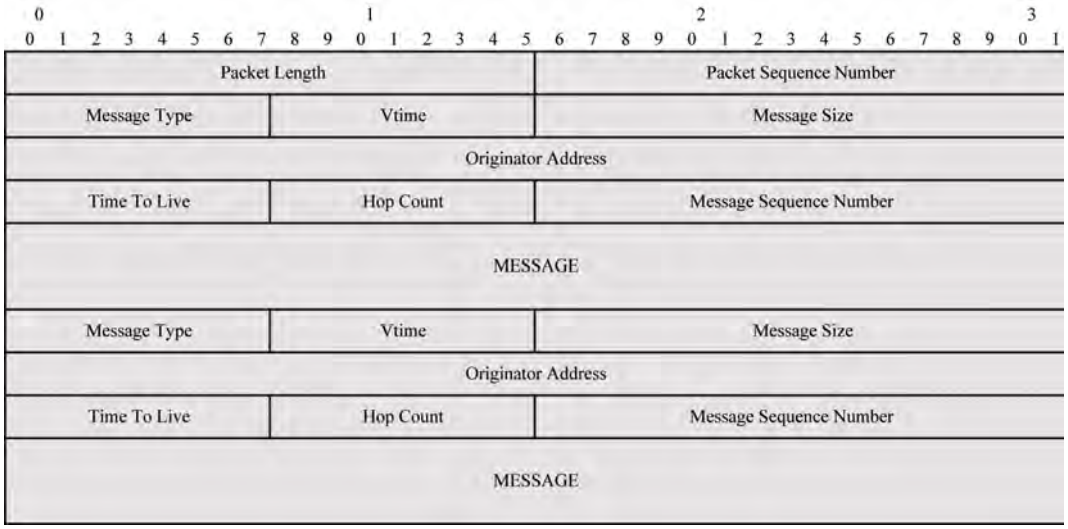


Figure 3.2: Formato del paquete OLSR

3.3.2.1 Packet Header

Figure 3.3 shows the fields of OLSR packet header:

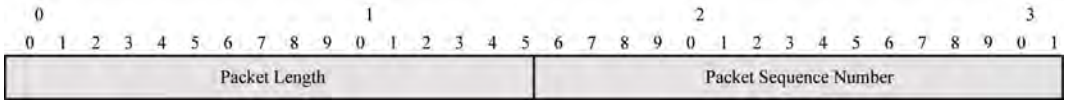


Figure 3.3: OLSR packet header

- *Packet Length*: [16 bits]. This field defines the size of the OLSR packet in bytes.
- *Packet Sequence Number*: [16 bits]. This field is used to define the packet sequence number. It must be incremented by one each time a new OLSR packet is transmitted.

3.3.2.2 Message Header

In the Figure 3.4 the fields of message header are represented:

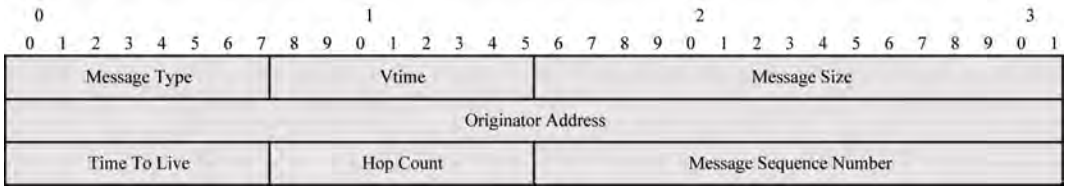


Figure 3.4: Message header

- *Message Type*: [8 bits].This field indicates which type of message is found in the MESSAGE field. The types in the range of 0-127 are reserved.

- *Vtime*: [8 bits]. Field that indicates the validity time of the information contained in the message.

- *Message Size*: [16 bits]. Size of this message in bytes, included the header and the MESSAGE field.

- *Originator Address*: [32 bits]. It indicates the main address of the node, which has originally generated this message. It should not be confused with the source address from the IP header, which is modified each time that the OLSR packet is retransmitted by an intermediate node.

- *Time To Live*: [8 bits]. It contains the maximum number of hops a message will be transmitted. Before retransmitting a message, the value of this field must be decremented by one.

- *Hop Count*: [8 bits]. The number of hops a message has carried out. It is set to 0, and it is incremented by 1 in each retransmission.

- *Message Sequence Number*: [16 bits]. When generating a message, the node that generated it assigns a unique sequence number which identifies each message in this field. The sequence number increments by 1 for each message originated by the node.

3.3.3 MID Message

If the node has more than one interface, this additional interface is periodically announced to other nodes using *Multiple Interface Declaration* (MID) messages.

3.3.4 Hello Message

The OLSR protocol uses regular exchange of *Hello* messages to discover neighboring nodes and the state of the network at link level. The message format is indicated in Figure 3.5 . As the main address of the node is included in the source address from header of the message, only the addresses of the additional interfaces have to be announced. *Multiple Interface Association Information Base* on the receiver node is built based on this information.

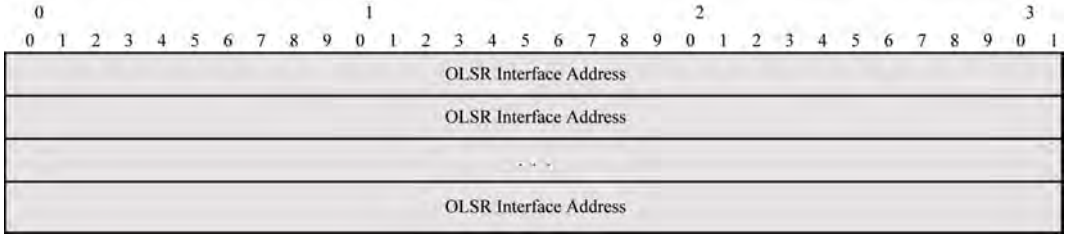


Figure 3.5: MID message format

3.3.4.1 Hello Message Format

Hello messages follow a format similar to the general packet, in way that they can include information for the detection of the link state of the network, to transmit signals for the detection of neighboring nodes, for selection of MPRs and to take into account future extensions.

The message format is indicated in the Figure 3.6 .

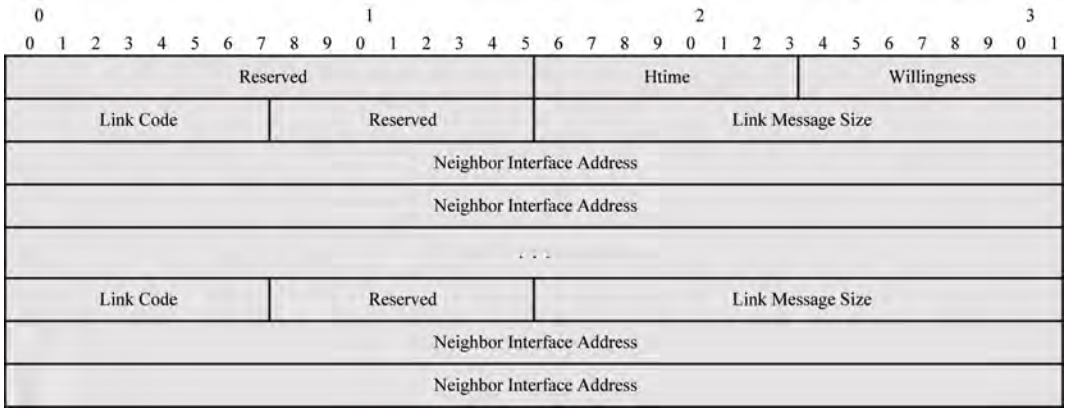


Figure 3.6: Hello message format

It is sent as data within the OLSR general packet described before, configuring the *Message Type* field with the HELLO_MESSAGE value and the field *Time To Live* (TTL) with 1.

- *Reserved*: [16 bits]. Reserved, it is set with the value 0x0000.
- *HTime*: [8 bits]. This field specifies the *Hello* emission interval used by the node, i.e., the time until the sending of the next *Hello*.
- *Willingness*: [8 bits]. It indicates the willingness which a node has to forward traffic for other nodes. If the availability of a node is defined as WILL_NEVER, this must never be selected as MPR node.
- *Link Code*: [8 bits]. This field specifies the type of link which sender node has with the neighbors in its list. At least, OLSR requires the following three kind of links:
  - *ASYM_LINK*: It indicates that the links between the sender node and its neighbors are of type asymmetric (i.e., it is only possible "heard" to the neighbor, but it is not possible to establish a bidirectional link with this).

- SYM_LINK: It indicates that the link between the sender and its neighbors are symmetric (A bidirectional link exists).
- MPR_LINK: It indicates that the nodes defined in the list have been selected by the sender as MPR.

- *Reserved*: [8 bits]. This field is reserved for future use, and must be set to 0x00.
- *Link Message Size*: [16 bits]. This field defines the size of the link message, which it is measured from the beginning of *Link Code* field until the next field *Link Code*. If the value of *Link Code* field does not exist, it will measure until the end of *Hello* message.
- *Neighbor Interface Address*: [32 bits]. This field defines the list of neighbors, which have been labeled as a *Link Code* in particular.

3.3.4.2 Hello Message Processing

The nodes processes received *Hello* messages for the detection of connections at link level, neighbor detection and MPR selection.

3.3.5 Neighbour Discovery

3.3.5.1 Detection of Connections at Link Level

Each node stores information about its connections at the link level with other nodes in a structure called Link set. With these connections, we refer more specifically to network interfaces using OLSR and its capacity to exchange OLSR packets. The mechanism used for this detection is the periodic exchange of Hello messages. To consider a valid connection we must check that there is communication (reception of *Hello*) in both directions.

Each neighboring node has associated a state in relation to the connection: *symmetrical* or *asymmetrical*. The first indicates that the two-way communication has been confirmed; and the second is used to indicate that *Hello* messages generated by the neighboring node have been received, but it has not been confirmed yet that the neighbor node is able to receive the *Hello* messages generated locally.

The confirmation of a neighboring node is able to receive the Hello messages emitted is got to find the own address in the Hello messages from the neighbor.

3.3.5.2 Neighbour Detection

Each node maintains a set of neighbor tuples based on the information about the connections stored in the *Link Set*.

Each neighbor tuple consists of data:

N_neighbor_main_addr ***N_status*** ***N_willingness***

where *N_neighbor_main_addr* is the main address of the neighboring node, *N_status* refers to the connection state (symmetrical or asymmetrical), and *N_willingness* is an integer between 0 and 7 representing the disposition or intention of the neighbor to relay traffic of other nodes.

Apart from the set of immediate neighbors or one-hop, which we have connection at link level, each node saves information about set of two-hops nodes of distance in the structure *2-hop Neigbor Set*. To this end, for each one-hop neighboring node, the set of

its one-hop neighbors is saved since they are announced in the *Hello* messages, which are received periodically.

3.3.6 Multipoint Relay (MPR)

The MPR are used to flood control message from a node to the whole network minimizing the retransmissions. Therefore, the concept of MPR is considered an optimization of the flood regular mechanism of messages in a network.

As shown in Figure 3.7 , each node in the network chooses, independently of the others, its own set of MPR among its one-hop neighbors with a symmetrical connection. The set of selected nodes is known as the MPR set for that node. The neighbors of node N that are not within the MPR group, receive and process the information of broadcast messages, but they do not relay the information coming from the node N.

The control traffic overhead generated by the routing protocol is directly proportional to the size of the set of MPR nodes in the network. At the same time, the MPR nodes maintain information about the set of one-hop neighbors which they have selected as MPR; this set is known as MPR selector set of a node. This information is acquired from the received *Hello* messages from one-hop neighbors.

Although the message pure flooding is more reliable and robust, this consumes a lot of bandwidth. The use of MPR nodes equally provides good results, with much less control traffic. Figure 3.8 illustrates a comparison, in terms of retransmissions, to get a 3-hops broadcast message in the network.

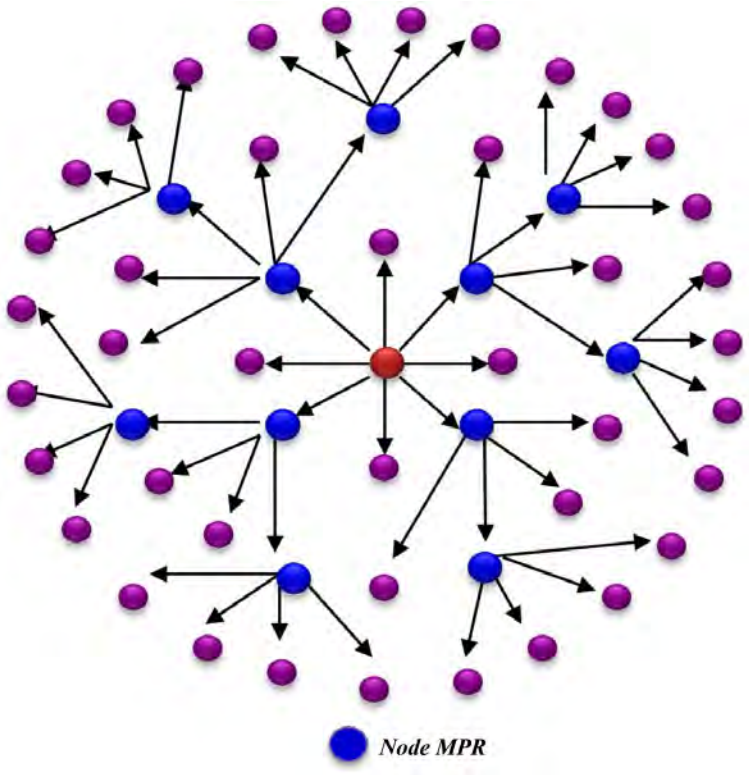
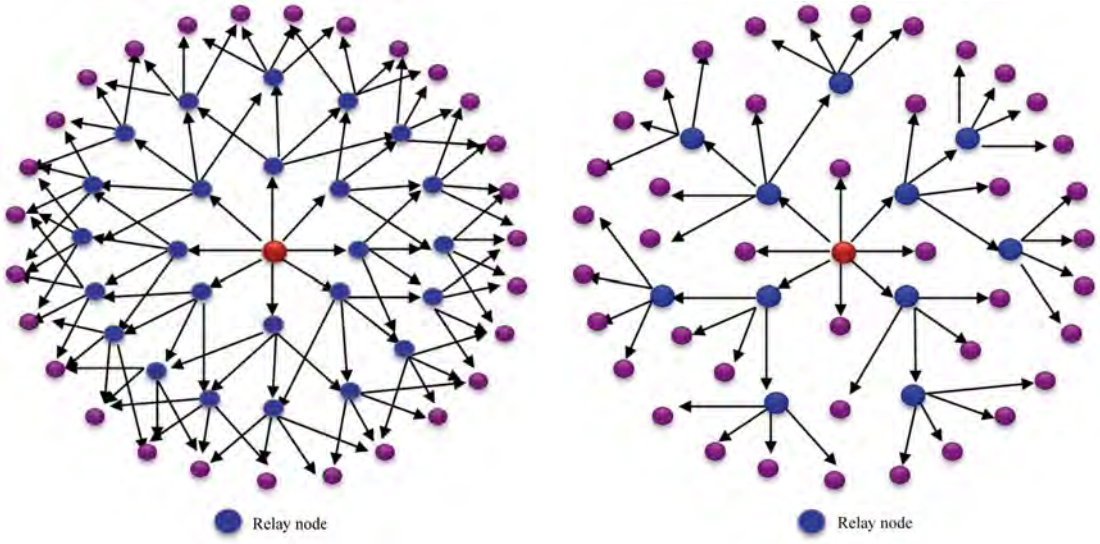


Figure 3.7: Selection process of MPR nodes



(a) Pure flooding (b) Using MPR nodes

Figure 3.8: Difference between pure flooding and the use of MPR nodes

3.3.6.1 MPR Selection

When the neighbor as MPR has been chosen, it is advertised in the *Hello* messages by setting the value MPR_NEIGH instead of SYM_NEIGH in the field *Link Type* before the neighbor address is chosen as MPR.

The MPR set is calculated by each node so that, through the selected neighbors, the node is able to reach to all two-hop neighbors. More specifically, to exact two-hops neighbors, so that one-hop neighbors are not taken into account. Although the MPR set should not be minimum to ensure correct functioning, no matter how smaller it is, we get less overhead produced by the control messages from OSLR protocol.

Each node also stores in the MPR selector set the set of nodes that have chosen it as MPR. They are detected while processing the received Hello messages.

3.3.7 Topology Discovery in OLSR

3.3.7.1 Functioning

The connection and neighbor detection provides each node with a list of nodes to communicate with directly and, making use of MPR nodes, a mechanism for optimized flooding. Based on this, information about the topology is generated and it is distributed over the network.

The TC messages is generated by nodes that have been chosen as MPR by some neighbor theirs. They serve to announce a set of links between the sender and other nodes, which is usually its MPR selector set, i.e., the neighbors which have chosen the sender node as MPR. These TC messages are emitted over the whole network by flooding.

Due to limitations of size of messages in the network, the announced address list can be partial in each TC message. However, to join all emitted TC messages all address of the MPR selector set must be found.

3.3.7.2 TC Message Format

The TC messages have the following format shown in the Figure 3.9 :

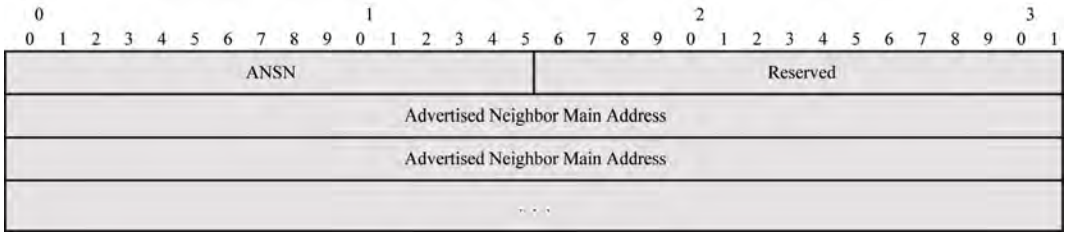


Figure 3.9: TC message format

This message is sent as data within OLSR packet, setting the *Message Type* field with TC_MESSAGE value and the TTL field with the value 255 (the maximum).

- *Advertised Neighbor Sequence Number (ANSN)*: [16 bits]. A sequence number is associated with the advertised neighbour set. Every time a node detects a change in the advertised neighbour set, it increments this number. It serves for nodes receiving this message to know if the information is the most recent to have occurred.
- *Reserved*: [16 bits]. This field is reserved. We set it to the value 0x0000.
- *Advertised Neighbor Main Address*: [32 bits]. Field which contains the main address of a neighbor node.

3.3.8 Routing Table Calculation

Each node maintains a routing table which allows it to route data, destined for the other nodes in the network. This table is based on the information contained in the neighbor nodes and TC control messages.

The entry format in this table is the following:

R_dest_addr	***R_next_addr***	***R_dist***	***R_iface_addr***
R_dest_addr	***R_next_addr***	***R_dist***	***R_iface_addr***
...			

Each entry means that the node with the address *R_dest_addr* is at a distance of *R_dist* hops from the local node, the neighbor node with the network interface address *R_next_addr* is the next hop in the route to *R_dest_addr*, and that this node is reachable from the local interface with the address *R_iface_addr*. An entry for each node in the network is maintained for which route is known.

Nodes with an unknown route are not included. The update of the table is carried out in case of emergence or disappearance of a node, either immediate neighbor, two-hop neighbor, or any other node known through TC control messages. The table is also updated when changing information on multiple interfaces that can be associated to the nodes.

This update of the table is an internal process that does not trigger the sending any message.

3.4 AODV Protocol

The AODV [PBRD03] routing protocol is an on-demand protocol based on distance vector routing. The nodes without any active route neither store routing information, nor participate in the exchange of routing tables. A node must not, therefore, discover and save a route to another node until it does not communicate with it, unless it is an intermediate node of two nodes that have established a communication. Each node maintains a routing table with information that has routes, so that it is not necessary that packets carry the route information to follow, with the consequent saving in bandwidth.

The routing table has a life time for each entry, so that if this time expires it restarts the search of a route to the destination that was associated. Similarly, entries in the table are associated with a sequence number that serves to avoid loops in the routes, as well as helping to distinguish old information updated later. The correct functioning of AODV predominantly depends on each node to keep its own sequence number updated.

3.4.1 Control Messages

Then the control messages defined in the AODV specification are described. These are known as AODV generic messages.

3.4.1.1 RREQ Message

Route Request (RREQ) messages are used by the nodes to begin the route setup process when they want to communicate with another node. To do this, they first increase its own sequence number and then flood the network with an RREQ message. The message format is shown in Figure 3.10 . During a period of time, the node will save the message identifier and its source address to avoid processing it if it comes back (see Figure 3.11).

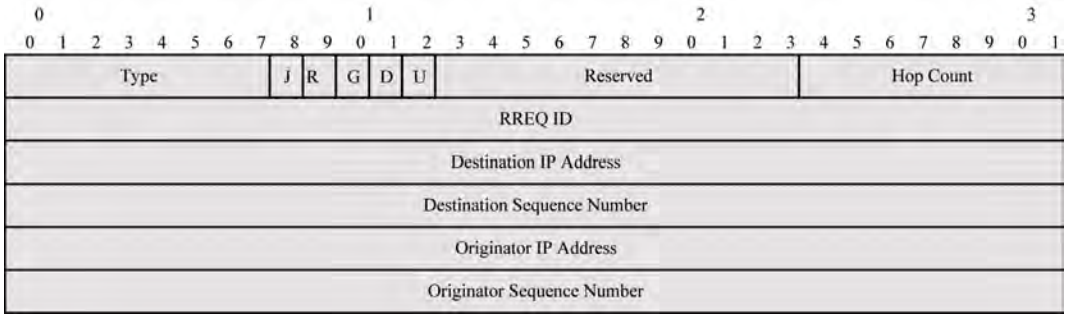


Figure 3.10: RREQ message format in AODV

To try to save bandwidth, the *technique of increasing the ring search* is used which consists of the sent RREQ having a minimum life time or TTL. In this way, only the devices near the origin node receive the RREQ. If there is no reply, it is going to increase the area where the RREQ messages are by increasing the TTL, up to getting to a TTL limit. This technique limits the propagation of RREQ messages and, in the case of not getting a response, the reach of the messages is increased.

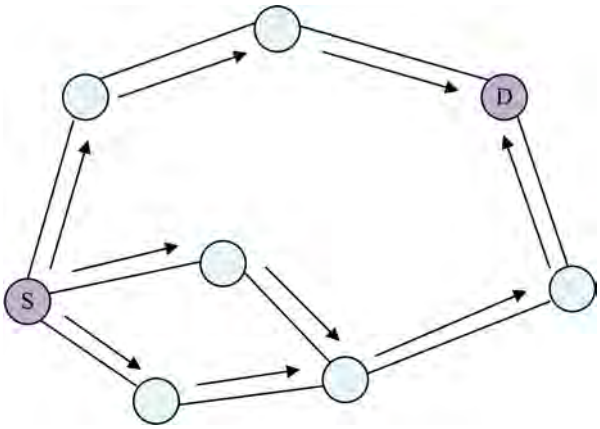


Figure 3.11: Propagation of a RREQ message in AODV

A node receiving a RREQ should create or update a route towards the neighboring node which has been transmitted it. Then it checks if it is a duplicated message and if this is the case, no more of the process is completed. If it is not duplicated the node creates or updates a reverse path toward the origin of the RREQ message. If such a route already exists, its sequence number should be updated with the one of RREQ message if the latter is greater. The "next hop" will be the neighbor from which the message has been received. For this route is allowed to relay the RREP if it comes back.

If the node that receives the RREQ is unable to generate a RREP it should relay the RREQ, before updating the sequence number for the destination of the message with its own number if it is greater than the one that carries the message.

3.4.1.2 RREP Message

Route Reply (RREP) are sent in response to the arrival of a RREQ, if the node is the destination or if it has updated information to reach such a destination. Up-to-date information is detected thanks to the sequence numbers. The RREP message format can be seen in Figure 3.12 .

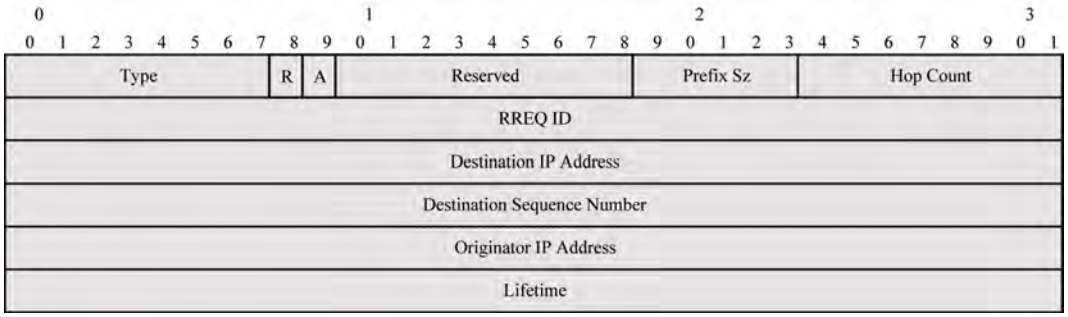


Figure 3.12: RREP message format in AODV

The RREP messages do not flood over the network but they are sent in unicast to the node that originated the route setup process by the reverse path created with the flooding of the RREQ (Figure 3.13).

If the node that generates the RREP is the destination, just before sending it, it increases its sequence number by one unit if that is the value announced by the RREQ. If it is an intermediate node, the one that triggers the RREP puts in the message, the sequence number that it has for the destination.

Nodes that processed a RREP create or update the route toward the neighbor which has sent it. In addition, they create or update the direct route to the destination (the sender of the RREP) to be able to route packets that are destined to that node.

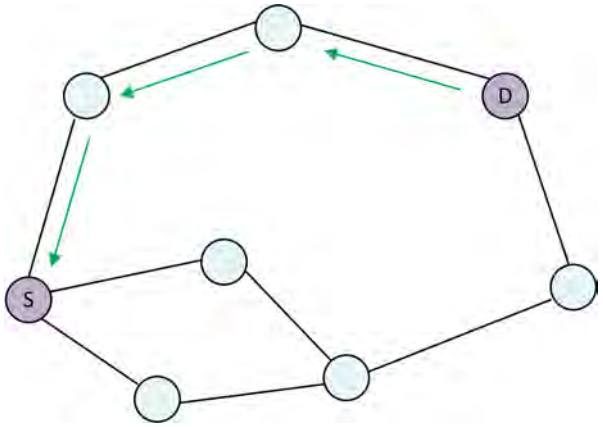


Figure 3.13: Back path of RREP message to origin

3.4.1.3 RERR Message

RERR messages are used to report that a specific destination cannot be achieved. The RERR message format is shown in Figure 3.14 .

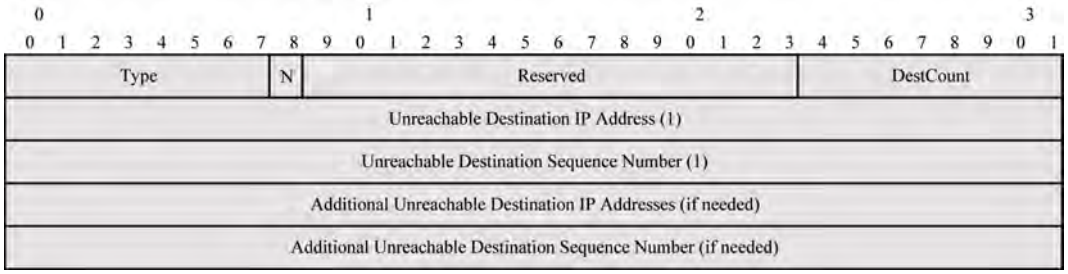


Figure 3.14: RERR format message in AODV

A node is not able to reach a particular destination due to the three distinct situations:

- When a node detects the loss of connectivity with a neighbor which is the “next hop” of an active route.

- When a node has to send a packet towards a destination where no active route is unknown.

- When a node receives a RERR of a neighbor announcing the loss of connectivity with neighbors that utilized with “next hop” in active routes.

3.4.1.4 Hello Message

The nodes that are part of active routes can send connectivity information to their neighbors through Hello messages. They are generated when in a certain period of time they have not transmitted any broadcast message.

Hello messages are really RREP with a TTL or life time of one-hop so they are only received by the neighbors of node which send them. Its format is shown in Figure 3.12 . When a neighbor processes a *Hello* message, it must create or update the entry in the routing table whose destination is the origin of the message. If some neighbor receives a *Hello* message from a node, and after a period of waiting time does not receive another message from it, it considers the link as lost.

3.4.2 Route Discovery

The route discovery is done by flooding the network with RREQ messages. When a node receives a RREQ creates or updates an entry in its routing table to the origin of the request, establishing in this way a reverse path to the node source to presuppose that the links are symmetrical. When the request arrives at the destination node it generates a response RREP message which transmitting to the source node by the established reverse path.

An intermediate node can also send a RREP, if it knows a more recent path. For that, the destination sequence numbers are used. To each new RREQ from, the source to a destination is assigned one higher number. Thus, if an intermediate node knows a route but it has a smaller number, it does not send the RREP. In addition to avoiding using old or broken routes, the sequence numbers serve to prevent the formation of loops that degrade the efficiency of the network.

Through the back RREP to sender by the reverse path, the route between the nodes source and destination is set. In turn, the sending RREP is used so that intermediate nodes for which the message is going to update their routing tables.

An optimization for this route discovery procedure is the technique of increasing ring search, as commented upon previously. It consists of sending the RREQ messages with a low life time (TTL) to prevent its spread throughout the network. If after some time, the RREP has not been received, another RREQ with one higher TTL is sent. This process of increasing the TTL will be able to be repeated until a TTL threshold is reached. Once completed, the network is flooded.

3.4.3 Route Maintenance

In routing table from AODV, the entries are distinguished depending on if they were created upon receiving a RREQ or a RREP. If they were created with the arrival of a RREP message, routes are forwarded, which are deleted if not used during a time interval of active route, i.e., if any data is not transmitted by this route, although the route remains valid. If the route was met by a RREQ message, it is said that the route is backward and it is eliminated by elapsing an interval of time normally smaller than the route active and broad enough to allow the return of the RREP.

When the link to the next hop in an entry from the table is broken, all active neighbors is informed. A neighbor of a node is considered active for an entry if a packet is sent by such an entry within the interval of active route. Link failures are propagated through route error messages, or RERR, which also update the sequence numbers of destination.

When a node cannot transmit a packet due to failure of the following link, it increases its sequence number of destination and generates a RERR including such a number. When

the source receives the RERR, it initiates a new route discovery toward the previous destination, but using sequence number at least as large as the one received. Upon arriving the new RREQ with the number given to the destination node, this sets it as its sequence number, unless it already has one number greater than the received. The neighboring nodes can exchange *Hello* messages periodically to detect link failures. No reception of this type of packets from a neighboring active can interpret it as the breaking of the link between them.

3.5 ZRP

ZRP [HPS02] is a hybrid routing protocol, since it combines the best properties of the proactive and reactive protocols.

ZRP is based on separating the network into areas. Clearly it is differentiated by a nearby area or neighborhood, composed of nodes that are a maximum of N hops, and the rest of nodes of the network.

ZRP uses two components to the routing: *Intra-Zone Routing Protocol* (IARP) e *Inter-Zone Routing Protocol* (IERP).

In the near area the component IARP is used which acts as a proactive protocol, keeping all the routes of the nodes that are to N-hops or less, being N a variable. The mechanism used for discover of the neighboring nodes is the periodic exchange of *Hello* messages.

ZRP has the component IERP for the global routing toward the nodes outside the area inside or nearby, which behaves like a reactive protocol.

When the route toward a new node is needed, the component IERP is used, and this route is requested through the mechanism *Bordercast Resolution Protocol* (BRP). This mechanism works by sending messages of route request to the nodes that belong to the frontier or border from inside area with outside area, i.e., to the nodes that are to the maximum number of hops from the interior area.

If any of these nodes from border know the route, they send a message indicating where the request came from. If not so, they forward the request over the whole network until a node is reached that knows a route to the destination. Then the response is sent back to the source, storing intermediate nodes through which the message can pass to be able to be used as a route to the desired destination.

As mentioned, the inner zone radius (in number of hops) is adjustable according to the needs of the network. As extreme cases, if this radio is small, at least one, ZRP will behave as a purely reactive protocol. On the contrary, if the radius is infinite, the behavior will be proactive.

3.6 Summary

The main objective of this chapter has been to show that sending a message from one node to another without a direct link existing, constitutes the purpose of routing protocols for mobile ad hoc networks. Also, various classifications have been presented of the same being the most relevant that which groups them under the procedure adopted for the path discovery to establish and to its maintenance and that divides routing protocols into three classes: proactive (protocols in which each node maintains information on how get to any other node in the network and exchange this information with all its neighbors) with low latency and high overhead, reactive (protocols in which a node only calculates a route to a destination when an exchange of packets is required with the same) with high latency

and low overhead and hybrid (protocols that combine aspects of the two previous, being proactive at the local level and reactive at global level) that minimize the disadvantages of both but at the cost of increasing complexity. Subsequently, we have seen an example of proactive protocol (OLSR), reactive (AODV) and hybrid (ZRP), describing in detail the first two since reference patterns are used to analyze the performance of routing protocols specified in the present document.

Chapter 4

Ant Colony Optimization (ACO) Algorithm

The overall objective of this chapter is to present the ant colony optimization algorithm, the theory which sustains the routing protocols developed in this Thesis. In the first place some basic notions are given. Then we analyze the so-called double bridge experiment. Following this, we present the artificial ants as computational agents that have very similar behaviour to that of the natural ants. Then, S-ACO variant is described, which finds the shortest path in a graph. Subsequently, it discussed some generalities of operation as well as various parallelization techniques of the ACO meta-heuristic similar. The chapter ends with a brief summary.

4.1 Introduction

ACO consists of a set of methods and techniques that are applied in generic optimization problems.

The ACO algorithm forms part of the so-called bioinspired algorithms, and within these, of those based on the concept of swarm intelligence, which applies the social behavior of insects and other animals to solve problems.

It is worth mentioning the collective of the ants. The ant as a unique individual has limited effectiveness, but as an integral part of a well organized colony, becomes a powerful agent that works the development of the colony. Ants live for the colony and exist only as part of it. Ants are able to communicate, learn, cooperate, organize and so on, and all together can carry out a specific mission.

There are a considerable number of researchers who have studied the behavior of ants in detail. One of the most surprising ant behavior patterns is the ability of certain species to find food by the shortest path. Biologists have demonstrated experimentally that they communicate by means of a chemical called pheromone.

The ants, in their need to find the food and bring it back to the nest, they explore a vast area and they indicate to others how to make the back journey to the colony. In this way, the ants know the path from their nest to their destination, without having a global vision of the field. Most of the time they find the shortest path and they are adapted to the changes of the land, showing its efficiency in this task.

This behavior pattern inspired to the researchers to develop optimization algorithms which help to overcome different routing problems. The first attempts appeared about the 90s and they have continued to evolve until our days.

4.2 Double Bridge Experiment

The Double Bridge experiment [GADP89] is to observe the behavior of a colony of ants of the Argentine species *Linepithema humile* (known as *Iridomyrmex humilis* until 1992) in the presence of the problem of finding a food source.

These ants, which are completely blind, realize the communication among them and their environment by means of a chemical substance called pheromone. The signal left on the ground is often referred to as *pheromone trail* and, logically, is fundamental for your social life. This type of pheromone is used by some species of ants to mark the route over the land and therefore they know the path towards the food and the back path to the nest. The ants smell the pheromone and they choose the route with the greatest pheromone concentration.

It has been checked that there are two types of behaviors: *assent of the pheromone trail and tracking* of the same. The cited experiment analyzes the behavior of ants seeking food on its trajectory from a point a (the nest) to another point b (the food). The experiment changes 3 times the ratio $r = l_l/l_s$ of the length of the branches of the double bridge, where l_l is the length longer of the branch and l_s the length of the shorter one.

- *First Case*: The bridge has two branches of equal length ($r = 1$). Ants leave the nest and they begin to move freely looking for a path which will lead them to food. In this way, they find two branches and a percentage of them choose a branch and another one selects another branch. They were observed during a specific period of time. The result was that, although in the initial phase, the choice was random, they continued choosing the branches in a similar percentage.

- *Second Case*: The relationship of length between the two branches is 2 ($r = 1$). In this manner, the long branch is double that of the short one. It was found that, in principle, the choice of path was random, as in the previous case, but after some time the ants chose the shorter branch. This has the following explanation: when the experiment starts there is no pheromone in any of the two branches. The ants do not have a preference and they choose the two branches with the same probability. As ants deposit pheromone while walking, the shortest path will be crossed more times, so after a while the amount of pheromone deposited in that trip will be greater, and as they choose the path with more pheromone, which it is the shortest one, they will cross over it since the greatest amount of pheromone encourages more ants to follow that path. However, all ants do not use the short branch, a small number continues along the longest branch. This can be interpreted as route *maintenance*.

- *Third Case*: We studied the behavior of ants when a single path is presented to them and once they are *accustomed* to it, we offered them another shorter path. That is, initially a unique path, which was crossed by ants for 30 minutes, was presented to them, and then we offered them a branch which gave them a shorter trip. The short branch was crossed occasionally, but the colony is held in the long branch. In addition, the high concentration of pheromone in the longest branch is maintained by slow evaporation. This behavior continues to support the longer branch, regardless of the fact that a shorter branch had been offered.

4.3 Artificial Ants

The artificial ants are agents that collaborate to solve computational problems. In our particular problem the artificial ant is a simple computational agent, which tries to

give solutions to the problem of calculation of the minimum path, exploiting the available trials of pheromone and the heuristic information. In some cases, it offers solutions that are unsuitable and are *penalized*, discarded or not depending on the error level of the solution.

In general, the artificial ant has the following properties:

- Search the *minimum cost* problem.
- It has an internal memory that stores information about the path followed until the time. This memory serves to:
  (i) Build valid solutions.
  (ii) Evaluate the generated solution.
  (iii) Rebuild the path which the ant has followed.
- It has an initial state with one or more stopping conditions (ended states).
- It starts in the initial state and it is moved building the solution incrementally.
- When a node is in a determined state and it has followed the sequence of visited nodes, it can move to any neighbor.
- The movement is carried out by applying a transition rule, which is in function of the deposited pheromone, heuristic values of the internal memory from the ant and the constraints of the problem.
- When an ant moves from a node to another one may update the pheromone trail associated with the arc between two nodes. This process is called *step by step pheromone trail online updating*.
- The construction process ends when some stopping condition is satisfied.
- Once the ant has a solution it can reconstruct the completed path and update the traces of pheromone trails from visited arches/components utilized a process called a *posteriori online updating*.

Artificial and natural ant colonies share a range of characteristics, since they interact and collaborate to solve a specific task. Here the most important ones are summarized:

- Both the artificial and natural ants modify their environment through a *stigmergic*[1] communication based on the pheromone. In the case of the artificial ants, pheromone artificial trails are numerical values that are available only in a local manner.
- The artificial and natural ants share a common task: the search for the shortest path (iterative construction of a minimal cost solution) from a source (nest) until a final state (food).
- The artificial ants build solutions repeatedly applying a strategy of stochastic local transition or probabilistic to move between adjacent states, similar to natural ants do.

However, these features alone do not allow developing efficient algorithms for hard combinatorial problems. The artificial ants have some additional capacities:

[1] Collaboration through physical environment

- Artificial ants may use the information heuristically to find the path which will lead them to the destination.
- They have a memory which stores the followed path.
- The amount of pheromone deposited by artificial ant is in function of the quality of the found solution.
- Another important difference is the moment of depositing the pheromone. Normally, the artificial ants only deposit pheromone after generating a complete solution.
- The pheromone evaporation in the ACO algorithms is different to as is presented in nature, since the inclusion of the evaporation mechanism is a fundamental issue to prevent the algorithm to stay stagnant in the first solution. Pheromone evaporation allows the artificial ant colony to forget its history slowly and to find new paths. This avoids us to fall into a premature solution.
- To improve the efficiency and effectiveness of the system, the ACO algorithms are enriched with additional capabilities such as the ability to look beyond the next transition (*lookahead*), the local optimization, the *backtracking* and the so-called candidate list, which contains a set of neighbors that may improve the efficiency of the algorithm.

4.4 Simple Ant Colony Optimization (S-ACO)

From the double bridge experiment Dorigo [Dor92] recreates the ant behavior to make an algorithm that finds the shortest path in a graph.

He starts considering a static graph $G = (N, A)$, where N is the set of vertices of the graph and A the set of undirected edges connecting them. He calls source and destination to the two points that it is wanted to set the shortest path, or as it happens with real ants, net and food.

The first problem that appears as a consequence of the pheromone updating is the creation of cycles. It can happen while the ants build the solution since, as they tend to go where there are more pheromone, they repeat the path that seems best to them. Even if the ants escape from cycles it would not be favoring the shortest path between the source and the destination anymore. Therefore, it seems reasonable to remove the pheromone update according to the moving of the ants. However, if this mechanism is suppressed, the system stops working, even for the simplest case of the double bridge. The reason is very simple: returning to real ants, the orientation of an ant is based on the trail deposited in the ground, in such a way that when it finds a food source, it should return to the net. If it does not mark its trail with pheromone, it is not able to go back, because it does not remember the path which it has followed to reach the food.

In S-ACO, Dorigo extends the basic capacities of the artificial ants by giving them an internal memory capable of storing the route that they have followed as well as the cost of the same. Thanks to this memory, artificial ants are able to implement several behaviors that allow them to build a solution efficiently:

- When the ant moves, it builds a solution based on pheromone routes probabilistically, without pheromone updating mechanism.
- It does a deterministic reverse trip with loop elimination, while the pheromone routes are updated.

- It evaluates the quality of the generated solution and, on the basis of it, determines the amount of deposited pheromone.

4.4.1 Functioning Modes

The artificial ants of S-ACO can operate in two modes: *forward* and *backward*. They are in *forward* mode when they move from the net toward the food and in *backward* mode source they return to the net.

When an ant in forward mode reaches the food source, it changes its behavior to *backward* mode and begins its back trip to the colony. In S-ACO *forward* ants build a solution by choosing in a probabilistic manner the following neighboring node to which it will move. The probability choice is influenced by the deposited pheromone previously by other ants in that arc. The artificial ants in *forward* way do not deposit any amount of pheromone while moving. This fact, together with the deterministic behavior of ants in *backward* mode, prevents the emergence of loops.

4.4.2 Path Search

Each ant collaborates in the problem solution, starting from the source node and makes decisions in each node. The information that each node stores in its neighbors is perceived by ant and it is utilized to decide the node which must go next in a probabilistic manner.

At the beginning of each process a constant amount of pheromone is assigned to all arcs. When the ant k is in the node i, uses pheromone trail τ_0 to calculate the probability of choosing the node j in the following manner:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha}{\sum_{l \in N_i^k} \tau_{il}^\alpha} & \text{si } j \in N_i^k \\ 0 & \text{si } j \notin N_i^k \end{cases} \tag{4.1}$$

where N_i^k is the list of available nodes which the ant k may go when it is in the node i.

In S-ACO a node contains the neighborhood of all the nodes that are connected to it, excepting the node from which the ant comes. In this manner, it prevents the ants going back to their origin node. Only in the case that the node neighbourhood is the empty set, it is permitted the ant comes over its steps. It should be noted that this procedure can induce the loop path generation in the graph easily.

4.4.3 Path Retracing and Pheromone Update

The use of an explicit memory allows an artificial ant to go back along the path that has led it to the food source. When an ant reaches its destination, it changes of behavior from *forward* to *backward* and it begins to build the back path. Before starting to build the back path, the ant eliminates loops that may be in the path that has built while it looked for the target node. The loop problem is that, while the ant does its return trip, it may receive the pheromone contributions several times, generating a phenomenon of loop self-reinforcement. While the ant returns, it deposits a fixed amount of pheromone on the arches that has visited. In particular, if an ant k in backward mode crosses the arc (i, j), it changes the amount of pheromone of the arc:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau^k \tag{4.2}$$

With this rule an ant utilized the arc that connects i and j increases the probability of using it by the rest of ants. The amount of deposited pheromone is a function of the number of ants that go cross over it.

In S-ACO the ant memorizes the path that has taken it to the solution together with the cost of the arches that it has travelled.

Thereby, it can evaluate the cost of the obtained solution and utilized it to adjust the amount of deposited pheromone in each arc that it travels, through a variable function relative to the cost of the path, so that with the shortest path search, we can get the best solutions quickly.

4.4.4 Pheromone Trail Evaporation

The pheromone evaporation may be seen as a mechanism that prevents the rapid convergence of the ants to a path that is not optimal.

In fact, the decrease of pheromone that is located in the path favors exploring new routes during the global search process. [DCD98b] points out that in the real ants this mechanism is also present, although it does not play a fundamental role. Suppose that it is not so, if there were not pheromone evaporation, ants would always follow the same path. Thanks to the evaporation process the ants change the exploration areas periodically. Precisely, this is the mechanism enabling them to survive and explore new areas and to have available other routes.

The evaporation mechanism in the natural pheromone (and in the artificial) plays a key role, because without it this system would not work well, as it has been seen in the third case of the double bridge experiment.

In the real ants, the pheromone intensity, that is present in the medium, decreases in function of the time. In S-ACO such evaporation is simulated by applying a rule of pheromone reduction, which is shown as following:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} \qquad \rho \in (0,1] \tag{4.3}$$

Pheromone evaporation makes best solutions are built every time since it is evaporated the pheromone that is associated with the first solutions.

4.4.5 ACO Metaheuristic

ACO constitute a metaheuristic. In other words, it is a heuristic method to solve a kind of general computational problem. It is often applied to problems that do not have an algorithm that finds a satisfactory solution or when it is not possible to implement the optimal solution.

ACO solves combinatorial optimization problems, how to find the shortest route. The distributed process to find the shortest route is an important source of research in artificial intelligence. The ACO algorithms work in an iterative manner. At each step all the artificial ants contribute to give a solution to the problem using so-called artificial pheromone matrix. Pheromone matrix is updated with the values associated to the found solutions.

ACO was applied at the first time to the known problem of the *Traveling Salesman Problem* (TSP) [SG07]: a salesman starting from a city has to visit N cities without repeating them and to return to the origin in the shortest possible time. In this problem is applied *Ant System* (AS) [DMC96]. In AS each edge has associated a value of artificial pheromone.

At the time of addressing a computational problem, utilizing this metaheuristic, several tasks or stages are identified:

1. Represent the problem as a set of components and transitions or a weighted graph that travels the ants to build solutions.

2. Define the meaning of pheromone marks to get decision in an appropriate manner. This is a crucial step in the implementation of an ACO algorithm and is not a trivial task.

3. Choose the heuristic preference of each decision that should take into an account an ant while building a solution. It is worth mentioning that the heuristic information is crucial to good performance when it is applied to local search algorithms.

4. Implement efficient local search.

5. Choose a specific and appropriate ACO algorithm.

6. Adjust the parameters of the ACO algorithm. A good starting point is to utilize configurations that have proven to be good in similar problems. Another possible alternative is to utilize automatic procedures of parameter setting.

The most important steps are the first four, since a choice little inappropriate in them, carries out the correction to be hardly by parameter adjusting.

4.5 Parallel Approach

ACO is a technique easily parallelizable by its distributed characteristics. Then the most representative works in the literature are commented upon.

[BKS98] constitutes the first parallel approach. This method presents limitations in its development when analyzing aspects such as the number of local iterations, the rules of assignment of tasks to processors, static/dynamic approaches and so on.

[MM98] introduces a new ACO parallelization technique to solve the *Shortest Common Supersequence* (SCS) problem, que tiene importantes aplicaciones en planificación de sistemas de producción, en ingeniería mecánica y en biología molecular. which has important applications in planning of production systems, mechanical engineering and molecular biology. It employs the *island model* with several colonies of ants that are separate and that exchange information according to the trail to follow, but instead of using a graph (typical representation in ACO) to represent the problem, utilizes a string of characters, assigning a pheromone value to each character of the same. The results show that this algorithm has a better heuristics than a genetic algorithm, but has the disadvantage that the functionality provided by the use of graphs is lost.

[Stü98] applies a *master/slave* approximation to parallelize the different search techniques from ACO solutions with the characteristic that they do not interact. Stützle employs a simple strategy to execute the independent and parallel sessions of an ACO algorithm. The empirical tests are performed with the MAX-MIN AS [DS04] to TSP demonstrate the efficiency of this approximation. However, it has the disadvantage that it depends on the problem itself as the available hardware.

[DKGG01] implement a new system of parallelization ACO for problems of industrial programming, testing it in a processor of shared memory with OpenMP. This technique improves the results of sequential approximation; increasing significantly the difference according to the execution time is increased.

[RL02] analyzes different strategies of parallelization that applies specifically to the Traveling Salesman Problem. These strategies are only a guide for the parallelization of

the ACO metaheuristic, and so it cannot be considered a formal and generic approach. The results show an acceptable *speedup*, noting that in complex problems is reached a better efficiency, but it has the disadvantage that it requires a great amount of information, not being scalable.

Finally, [TG09] performs a test to evaluate the performance of the communication *Message Passing Interface* (MPI) multithreading. In this approach, hybrid programming models can be used, combining MPI across nodes and *multithreading* within a node, because many MPI implementations are beginning to support multithreaded MPI communication. With this technique, best results are obtained by nodes interacting in a more efficient manner.

4.6 Summary

The objective of this chapter has been to introduce the Ant Colony Optimization algorithm, briefly known as ACO metaheuristic, which is particularly suited to solve difficult problems of combinatorial optimization. This meta-heuristic method consists of a set of agents (ants) artificial that they cooperate with each other by means of a set of rules that determine the generation of local and global information and its update in order to find the best solutions. It has begun analyzing the so-called the Double Bridge experiment to show the main similarities and differences between the natural ants and the artificial ants. Subsequently, the S-ACO algorithm has been described, which has its inspiration in this aforementioned experiment and that solves the problem of finding the shortest path in a given graph. Then, we have commented the way of addressing a computational problem through this metaheuristic. Finally, various parallelization techniques that take advantage of the distributed nature of this algorithm have been exposed.

Chapter 5

Adaptive Routing

The overall objective of this chapter is to review the most representative works about adaptive routing protocols for mobile ad hoc networks existing in the literature. In first place we introduce the notion of adaptive routing. Then, ACO routing is presented as a particular type of adaptive routing. Then, it is noted the peculiarities of ACO routing in the case of its application to mobile ad hoc networks. Then it is described the main ACO routing protocols for mobile ad hoc networks, emphasizing AntHocNet, referenced protocol in the area. The chapter ends with a brief summary of the exposed in the same.

5.1 Introduction

It is so-called adaptive routing in a network to the set of techniques or routing protocols which, as its name suggests, try to *adapt* to the variability of the same (traffic, topology and so on).

Within this, we make special mention of the so-called ACO routing or set of routing protocols that make use of the ACO techniques.

The ACO approach for routing is quite robust because the loss of ant/s is not important. This approach differs from the approach of distance vector, where routing information comes from information provided by the neighboring nodes and link state approach, where the routing information is updated with received messages from the nodes in the network.

5.2 ACO Routing

An essential aspect of the ACO routing is that the ants always show various full paths between the source and destination, increasing the overhead respect to a purely reactive approach.

Another characteristic is the way in which the ants choose the route. They construct the path hop by hop in a probabilistic manner using pheromone information. The use of this allows it to build on the acquired experience by the ants previously. This is the key to a highly distributed process. The fact that ants build their paths in a probabilistic manner allows the exploration of multiple routes. This makes the algorithm to adapt to changes in the network, increasing both the robustness (through the availability of reserve paths) as the *throughput* of the network.

A third characteristic is the stochastic forwarding of data packet based on the pheromone information, which ensures its routing by the best routes. If the pheromone keeps up-

to-date by the use of enough ants, the load balancing follows changes in the network automatically.

All the previous make routing ACO algorithms to present very interesting properties:

- They work in a completely distributed way: the information is not in a central node, but it is contained in each node.
- They have great adaptability to the network and traffic changes.
- Use mobile agents (ants) to determine routes for sending the data. These agents are control packets that are sent over the network. There are two types: forward ants (they go from source to destination) and backward ants (they go in the opposite direction).
- They may provide multipath routing.
- They present an excellent fault tolerance, that is, they offer a good behavior to the failure of the agents.
- They choose the route for sending the data automatically.

5.3 ACO Routing in Mobile Ad hoc Networks

The direct application of the ACO algorithm as it is described by Dorigo [Dor92, DS04] is not advisable in mobile ad hoc networks by the slow convergence that it offers (when it is the case). His proposal is developed in a static network topology, in which all routes are known in advance. What ants do is choose the route based on traffic load. In mobile ad hoc networks, where the topology is dynamic, the routes are not always valid. A dynamic topology implies that the routes for the communication are unknown. The first step is, therefore, to do an exploration [DDCG10] that captures the topology of the network quickly and without much cost. [Gor00] shows that the exploration carried out by collecting ants is *directed*: some certain ants, which are part of what is called the *patrol*, go out in several directions to explore the round of the colony. These ants, upon returning to the colony, indicate whether they have found food or not. If so, they stimulate somehow collecting ants to go out toward the food in the indicated direction by them, creating a flow of traffic between the net and the food. Subsequently, collecting ants obtain the minimal route, but only in the indicated area by the patrol. The calculation problem of minimum route is, therefore after the exploration. ACO works similarly. By a patrol similar mechanism, it is got to the target in a local area. Then, searches in depth in such an area are performed.

5.4 Related Work

AntNet: Distributed Stigmergetic for Communications Networks [DCD98a] is the first ACO routing algorithm for networks, in this case for wired networks or static (as not dynamic). It is multipath and is adapted to the network traffic, not being necessary exhaustive routes calculation. It does not require maintenance of the same either and it does not need to update global information because in the case that a route may suffer modifications is the same node which performs this task.

AntNet-FA (*Flying Ants*) [DCD98b] is an enhanced and scalable version of AntNet, in which the forward ants make use of high priority queues similarly to the way the

backward ants do. The latter update the routing tables at each visited node using local estimates of their traveling time, and not the experienced by the backward ants. AntNet-FA performance improves with the size of the network and its efficiency is similar or even better than the one of AntNet.

AntNet and AntNet-FA are not ACO routing protocols for mobile ad hoc network but they are, without a doubt, their predecessors.

Then we describe the most important ACO routing protocols more important, grouping these in proactive, reactive, and hybrid.

The most representative *proactive ACO routing protocols* are the following:

Adaptive Swarm-based Distributed Routing [KESMI+02], better known as Adaptive-SDR, is inspired in AntNet. This protocol has the property of grouping the nodes in colonies to solve the problems of scalability of other protocols, derived from the fact of each node has to send an ant to others. The grouping of nodes in the colonies is done by a control central entity that becomes aware of their geographical positions. There are two types of ants: Colony ants and local ants. The first ones have the mission to find routes from one colony to another. The local ants are inner to the colony and they find routes within the colony, relying on two routing tables. This protocol has many drawbacks: many colonies are not advisable due to the overhead that cause, to know the optimal number of nodes that should be in a colony is not anything trivial, in distributed systems is not always had a control central entity as it presupposes, large processing of routing tables carried out by the local ants implies a high resources consumption and it requires devices with important performances and so on.

Mobile Ant-Based Routing [HB03], better known as MABR proposes a scheme to tackle the scalability problem of the routing in mobile ad hoc networks. This approach abstracts the network dynamic topology to get *logical routers* and *logical links*. These two concepts relate to the set of nodes and created paths among them, respectively. This algorithm uses the geographic partition of the area of the node and the geographical addressing of pheromone exploration. In this protocol the forward ants checked regularly if a path to a randomly chosen destination is functional and the backward ants reflect the current state of the network, making sure that paths followed by the ants are reinforced positively or negatively. In addition this approach uses the pheromone evaporation, which favors more exploration and eliminates out-of-date paths. The problem with this proposal is only limited to present the theoretical model, not providing experimental results.

Probabilistic Emergent Routing Algorithm for mobile ad hoc networks (PERA) [BM03] is a protocol that adjusts at each node the likelihood that each of its neighbors can receive and forward the data packet. Each forward ant contains the IP addresses of the source and destination node, a sequence number, a field hop counter and a stack that it grows dynamically. The stack contains information of nodes that the forward ant visits as well as the associated times. When a node does not have a record of a route to a destination, it is created an forward ant where the node puts its own IP address into the ant stack, as well as the time in which the ant was created. From this moment the node stored forward ants sent to destinations when the route is required periodically. When this forward ant reached the destination, the destination node creates a backward ant. This new agent uses the information contained in the forward ant in the opposite way to update the probability distribution at each node, and to reflect the current state of the network. The fact of forward ants is sent in broadcast mode from the source and intermediate nodes causes a multiple broadcast for finding different routes to the destination, causing a great overhead.

Ant Routing Algorithm for Mobile Ad hoc networks (ARAMA) [HS03] is a proactive

routing algorithm in which the forward ants not only take into account the factor of hop counter (like most of the protocols), but they also value the local link heuristic through the route (such as the battery energy of the node and the queuing delay). The algorithm defines a value called grade. This value is calculated by each *backward* ant based on the information stored in the forward ant. At each node, the backward ant updates the amount of pheromone of the routing table from the node using the *grade*. The protocol uses the same *grade* to update the pheromone value of all links. The authors claim that the overhead of route setup and maintenance is reduced by controlling the number of forward ants. However, they do not clarify how to control the generation of ants in a dynamic environment.

AntNet Ring Search and Local Retransmission (AntNet-RSLR) [RMH11] is an adaptation from AntNet to mobile ad hoc networks through the incorporation of two techniques: *Expanding Ring Search* (ERS) and *Local Retransmission* (LR). In this protocol the mobile agents build paths between pairs of nodes, exploring the network concurrently and exchanging obtained information to update the routing tables, allows it to reduce as the overhead as end-to-end delay with respect to AntNet, AODV and DSR. Using the technique of the expanded ring searching the request message of route setup is spread progressively by flood from the source node. Initially it is spread the message to a small neighborhood with a small TTL value, which is going to increase until you reach the destination. This message is forwarded by the source node if any response is not received at a time interval. If the route request the TTL value has reached a certain threshold without receiving a response, it assumes that the destination is unreachable. However, this produces a great overhead and can cause loops that reduce the delivered packet ratio. To solve the problem of overhead, it introduces a variant of this technique called *Blocking-ERS*, which does not assume the route search procedure from the source node when a new sending in broadcast mode, generating a *rebroadcast* from an intermediate node chosen conveniently. Local retransmission technique is used when an intermediate node does not receive the corresponding data packet by expiating the timer value, sending a negative notification control menssage (*Nack*) for the intermediate node (and not the source node) returns to retransmit the data packet failed. This has the disadvantage that is unknown a priori buffer capacity from the node that stores the data for its possible retransmission. Upon being based on a proactive protocol as is AntNet, the overhead should be present as a negative aspect, although the authors claim that it is reduced.

The most representative *reactive ACO routing protocols* are the following:

Ant-colony based Routing Algorithm for mobile ad hoc networks (ARA) [GSB02] is reactive protocol in which the entries of routing table contain pheromone values that facilitate the choice of neighbor. To get a destination is necessary choosing a neighbor that serves us as link and so in turn until reaching to the destination. In the routing table pheromone values are degraded over time and the nodes enter *sleep* mode if they reach a certain threshold. The route setup is performed by flood, i.e. forward ants are forwarded to their neighbors. Each node receiving this ant updates its routing table. The duplicated forward ants are identified through a unique sequence number and they are removed. Upon receiving forward ant, the destination node extracts its information and it creates a backward ant, which returns to the source node. These have a similar task to forward ants. It should be noted that flooding has longer range than the broadcast since the packets are transmitted to all nodes in the network nodes via multihop, while packets by broadcast are transmitted only to neighbors that are one hop. The flooding problem is the high overhead involved. Once the route setup was made for a specific destination, the sender node does not generate a new mobile agent toward destination

anymore, but route maintenance is done by the data packets. The authors affirm that, in the considered scenarios, the performance of this protocol is very similar to the DSR presenting less overhead. However, it includes neither the scenarios representing a high network load nor multimedia data.

Ant-based Distributed Routing Algorithm for ad-hoc networks (ADRA) [ZGL04] is a reactive algorithm in which the ants move through the network between pairs of nodes chosen randomly. These ants move depositing pheromone based on several parameters: distance in hops from its source node, the link quality, the found congestion in their travel, the current pheromone which node has and the node speed. Of course, the same node, by the pheromone evaporation, changes its value according to the link age. An ant selects its way at each intermediate node according to the pheromone that the node has distributed and to speed up the choice of the way parameters with different values are given that update the probability in the routing table. The authors assert that ADRA presents less average End-to-End delay, a lower overhead and a better delivered packet ratio than DSR. Also, it allows them to optimize various QoS parameters, such as link quality, node load and so on.

Ant Colony Based QoS Aware Routing Algorithm for MANETs [LF05] is a link-disjoint multipath reactive routing algorithm. Most of the protocols are essentially routing methods of single route, which tend to have an overhead in the nodes that are in the shortest path from the source to the destination. This overhead is due to unique path methods there is not a load balancing. Link- disjoint multipath routing is more robust, supporting better QoS. It sets and utilizes link-disjoint multiple routes to send data packets and to adapt the pheromone to disperse the communication traffic.

A trend that has gained strength in recent years is the design of special single-path routing protocols. This type of protocols has the disadvantage that, having a unique path, if there is a link rupture there will not be another alternative, being a new route setup needed again with the resulting delay, with control message overhead and the decrease in the delivered packet ratio. To improve these problems emerges *Efficient Ant-based Routing Algorithm for MANETs* [WSJX07]. This protocol has been designed to allow more paths to be in packet request/response routes and to discover them with a lower overhead.

Position Based Ant Colony Routing Algorithm for Mobile Ad Hoc Networks [KO08] is a reactive routing algorithm based on the location of the nodes. The algorithm finds optimal or closer routes in a network that contains nodes of different transmission ranges. Each node assumes its position, the position of its neighbors, and the position of the destination node. Only it calculates the route at the time of sending the data from a source node to a destination node, and these data are sent when the route is set. To minimize the time that the algorithm needs to find a route, the information about the position of the nodes is used as a heuristic value. The use of the location information as a heuristic parameter reduces the needed time to establish routes from the source to the destination significantly as well as the number of generated control messages. It has a high delivered ratio and a low packet average delay packet, compared to other routing algorithms based on positions. The algorithm is stabilized before AntNet.

Ant Routing Algorithms in Ad Hoc Networks with Critical Connectivity [RBR08] is an algorithm that creates routes on demand in order to reduce the routing overhead with respect to proactive approaches. The forward ants only collected information of nodes that they cross over, choosing the next hop toward the destination only based on the amount of pheromone. The amount of deposited pheromone by the *backward* ants at each travelled link is constant. The simplicity of the protocol is useful to make a transparent routing in networks consisted by heterogeneous elements. In the algorithm at each node the routing

tables are probabilistic: the next hop is selected in accordance with the highest percentage of pheromone left by the ants. Thus, the forwarding of the forward ants is probabilistic and it allows the exploration of other routes available in the network. Data packets are sent in a *determined* (unicast) manner by intermediate nodes that are in the path from the sender to the destination node. This process creates a global route through the use of local information.

The *imProved Ant Colony Optimization routing algorithm for mobile ad hoc NETworks* (PACONET) [OTT08] is a reactive routing protocol where the forward ants explore the paths in the network in search of paths from a source to a destination in restrictive broadcast mode and the backward ants established the acquired path by the forward ants. Data packets are sent to nodes that have the highest pheromone concentration probabilistically. The forward ants travel toward unvisited nodes, but they do not find them, they follow the path of the node with the highest pheromone concentration. The rows of the routing table represent the neighbors of a node and the columns represent the nodes in the network. Each pair (row, column) in the table routing has two values: (a) a binary value that indicates whether the node has been visited, and (b) the pheromone concentration. The forward ants take only into account the pheromone concentration once all neighbors in a column have been visited. The purpose of this is to ensure that all paths are scanned to find the best route to the destination. The node with the greatest pheromone is chosen as the next hop, after forward ant has determined that the node has not been visited before.

The most representative *hybrid ACO routing protocols* are the following:

Mobile Agents based Routing Protocol for Mobile Ad Hoc Networks [MTS02], better known as Ant-AODV, is a hybrid routing form based on ACO and the AODV routing protocol. To overcome some of the disadvantages of AODV, such as the generated overhead by the increase of control messages, this hybrid technical is utilized which highlights the node connectivity and reduces End-to-End delay, as well as the route setup latency. The Ant-AODV ants work independently and provide routes to nodes. The nodes also have the ability to perform a route setup on demand for destinations that do not have a route entry the sufficiently updated. Use of ants with AODV increases nodes connectivity, which is associated with the number of destinations that have such a node and which can be reached by means of the corresponding entry in the routing table. Before route RREQ request, the likelihood of receiving a faster response is greater by having more connected nodes. As the ants update the routes continuously, a source node can select a new route and shorter. This leads to a considerable reduction in the average End-to-End delay, compared with AODV and with the routing based on ACO.

In addition Ant-AODV uses (RERR) path error messages to report on chain to other nodes of local link failure in a similar way as AODV does.

AntHocNet [DC04][DCDG04] is an adaptive, multipath and hybrid ACO routing algorithm. Data from 2004 and in these almost ten years has had many extensions and variations to improve their performance. As mentioned above, AntHocNet is an algorithm hybrid (reactive and proactive), multipath and adaptive. It is reactive because it has agents operating on-demand to set up routes to destinations. It is proactive because it has agents collecting information and these agents can discover new routes which serve as alternatives if a link fails It is a multipath because it provides different routes to send information to the destination. Finally, it is adaptive because it adapts to traffic conditions.

AntHocNet follows a structure similar to AntNet-FA, but it differs in its characteristics. As has been seen above, AntNet-FA is applied to topologies of static networks, in which the route are known and convergence is slow. So, what all ants have to do is choose

the path. AntHocNet, meanwhile, it takes into account the dynamic topology and other characteristics of ad hoc networks. When the network topology changes, then it must be restored quickly and this is achieved through a new route setup process. If multiple resources are used to accelerate this process, the exchange of information is enhanced. This can cause the network to collapse. Therefore, we have a problem: if we do not want to overhead the network, we increase the convergence time of ACO algorithm; if we want to reduce the convergence time, we overload the network. In other words, AntHocNet is a modification of AntNet-FA algorithm that it accelerates its convergence time without getting overhead the network. The main challenge of the MANETs, when applied to routing schemes based on ants, consists of finding the correct balance between the rates of agent generation and the the associated overhead. The adaptive property, which occurs in AntHocNet, it is suited to traffic and network conditions.

In the behavior of AntHocNet, the following phases are distinguished:

i) Routing information setup: It starts with the sending, on-demand, agents for the calculation of the path to the destination. This phase is supported by property multipath, which is considered of great importance, since they have to create the routes as soon as possible, so that the least number of data packets are lost.

ii) Data routing: In this phase, data is sent in a unicast and stochastic manner using the pheromones of the routing tables which have all the nodes locally. This routing strategy aims to expand the data load getting a better load balancing.

iii) Path maintenance and exploration of another new one: In this phase, while a data session is ready to relay the information, it is sent maintenance proactive ants according to the data sending rate. The purpose of this phase is to upgrade the quality of the route links and the values of pheromone between the path that goes from the origin to the destination.

iv) Management of link failures: In this phase the nodes can detect link failures. Once they are detected, AntHocNet try to mitigate using different mechanisms like sending messages of failure notification and the route local repair.

The simulations analyses different scenarios in term of number of nodes, mobility and density of nodes. In general, in all of them, AntHocNet has similar behavior or even better than AODV. In particular, it is obtained better delivered packet ratio than AODV, a End-to-End delay is slightly higher in AntHocNet than in AODV for most simple scenarios (high density and short paths), improving in AntHocNet for more complex scenarios. Finally, AntHocNet also has a good performance for large networks, which is a scalable protocol.

A unicast routing protocol for mobile ad hoc Networks using Swarm Intelligence (ANSI) [RS06] presents a protocol in which the routing table contains an entry for each node that is achieved and the best next hop, while decision tables of the ant store the values of pheromone. In this protocol the forward ants are only generated when a node needs to transmit data to another node. These ants are sent in broadcast mode, while the backward ones are sent in unicast mode, following the trail that the forward ants have left along the path and updating the values of pheromone of the nodes. The data packets choose the next hop, taking into account the highest value of pheromone. The protocol is as good or even better than AODV with regard to the packet delivery and End-to-End delay.

The Ducatelle's Thesis [Duc07] is an evolution of AntHocNet, regarding the Di Caro's Thesis [DC04] and other works such as [DCDG04]. The differences between both versions

are in the use of different mechanisms in the route reactive setup process and the route maintenance proactive process.

With regards to the setup process, older versions create multiple routes in this process. With this strategy, we have the advantage that multiple routes are available from the beginning of the data session, so that the session is better protected in face of the link failures and it can start the load balancing immediately. However, this may lead to a high overhead that sometimes is unnecessary. This last is experienced then deciding to create only a route in the setup process and to obtain multiple paths in the route maintenance process.

With regard to the route maintenance process, the predecessor versions consist on sending of proactive ants for the route exploration, not applying the pheromone diffusion process. These proactive ants are sent, therefore, in unicast mode and also in broadcast mode with some probability, because it has to explore new routes without the use of pheromone diffusion. Although overhead is reduced for not to use pheromone diffusion (*Hello* messages are more simple using less memory in bytes), the sending in broadcast mode causes a blind exploration completely because the proactive ants are sent to many nodes unnecessarily. After analysis of these two approaches we check finally the maintenance process provided by Ducatelle's version is more effective and efficient than its predecessors.

[WDR08] introduces several modifications to AntHocNet in the route setup phase to control the number of ants moving over the network and conveniently to update the values of pheromone in all intermediate nodes when the route is set. The simulations demonstrate that it reduces the overhead and End-to-End delay, while the delivered packet ratio keeps equal compared with AntHocNet.

[KD08] includes in AntHocNet the disjoint-node multipath property. The protocol facilitates balancing load especially and a lesser extent, fault tolerance, and the reduction of End-to-End delay. Multiple routes generated by AntHocNet are not disjoint. These routes may have links and nodes in common, presenting disadvantages with respect to disjoint routes providing this algorithm. Disjoint link or node routes are those that for one same data session do not share nodes or links, respectively. The existence of these disjoint node routes allows the load to be better distributed.

[DDCG08] is a variant of AntHocNet whose proactive part runs in the background offering a *best effort* routing service and the reactive part provides a connection-oriented service. Its most important property is that it can choose between proactive and reactive routing for each data session separately. Data packets can follow as paths proactive as reactive. There is a synergy (interactive cooperation) because the reactive part of the algorithm relies on proactive routing information. The simulations show that this variant improves the results obtained by its predecessors.

[DCDG08] realizes the performance study between AntHocNet and AODV in an urban environment using real-time applications. It is done a realistic simulation in terms of propagation radio, restriction of the node mobility and data traffic. In other words, the scenario is real with obstacles and real traffic patterns. QualNet is used for the simulations. In the majority of scenarios AntHocNet improves than AODV in terms of delivered ratio, delay and jitter. In addition, in the majority of tests, AntHocNet also has lower overhead than AODV and OLSR. In urban scenarios, AntHocNet has the advantage that the local density (number of neighbors) experienced by each node is relatively low and grows slowly. It is observed that the number of nodes affects much the delivered ratio, while the node movement does not seem to have as much influence. Also it is observed that the density of nodes has a strong impact on the delivered ratio, while the node speed seems to have

relatively minor impact.

AntHocNet [DC04, DCDG04] and all its variants [Duc07, WDR08, KO08, DCDG08] are a benchmark in the area of ACO routing protocols for mobile ad hoc networks. Their adaptive nature means their performance metrics are the best overall. Nevertheless, it presents some scalability issues in highly dynamic scenarios.

HOPNET: Hybrid ant colony OPtimization routing algorithm for mobile ad hoc NETworks [WOTT09] is a hybrid routing algorithm based on ants hopping from an area to another. The algorithm has features extracted from the ZRP and DSR protocols, being highly scalable protocols compared with other hybrid protocols. Proactively, this algorithm discovers the route within the vicinity of a node area, doing communication between areas in a reactive form. The size of the zone is determined by the average length radius in hops and not the node. Therefore, the routing zone is constituted by the nodes that are within the specified length radius. A node can belong to multiple zones that are overlapped and areas can vary in size. The nodes can be classified as interior and boundary (or peripheral). Boundary nodes are at the distance of the radius, while others that are less than the radius are interior. Each node has two routing tables: *IntraRT Intrazone Routing Table* (IntraRT) and *InterRT Interzone Routing Table* (InterRT). The IntraRT is proactively maintained so that a node can quickly obtain a path to any node within its zone. This is done by periodically sending out forward ants that detect the paths within the zone and topology changes (such as nodes moving away, link failure, new nodes entering the zone and so on). When a forward ant reaches a destination, a return ant (backward) is sent along the path discovered. The InterRT zone stores the path to a node beyond its zone. This routing table is setup on demand and the peripheral nodes are responsible for linking the zones. When the number of nodes is small, the continuous movement of the peripheral nodes does to have to discover new routes constantly, causing more delay than in other hybrid routing protocols.

Hybrid Routing Algorithm Based on Ant Colony and *ZHLS Routing Protocol for MANET* [RAP10] is a protocol that combines ideas of ACO with Zone-based Hierarchical (ZHLS) protocol. It works similarly to HOPNET [WOTT09], behaving in a proactive manner within a zone and carrying out a communication between different zones in a reactive manner. The authors assert that this protocol presents a better delivered packet ratio, a less overhead and a minor End-to-End delay than traditional algorithms.

5.5 Summary

The main objective of this chapter has been to understand more about adaptive routing protocols for mobile ad hoc networks based on Ant Colony Optimization algorithm. This review has been initiated with AntNet and AntNet-FA protocols, ACO routing protocols for wired or static networks that are the precursors of different ACO routing protocols for mobile ad hoc networks. Subsequently, we have analyzed in chronological order the most representative of the literature, dividing this tour in proactive, reactive and hybrid protocols, with emphasis on the latter and, particularly, in AntHocNet, multipath hybrid routing protocol which constitutes the immediate antecedent of the developed work in this Thesis.

Chapter 6

Adaptive Routing Protocols for Mobile Ad Hoc Networks

This chapter presents the design and specification of an ACO routing protocol family for mobile ad hoc networks. This set parts of a base protocol so-called optimized ACO routing protocol or, more commonly, AntOR. AntOR presents two main variants: AntOR-DLR and AntOR-DNR. In turn, from each of these are precursors of different protocols. The chapter begins by pointing out the most important aspects of AntOR, its phases and the main differences with respect to AntHocNet, protocol which is inspired AntOR. Subsequently, it describes the data structures utilized by this protocol, structures which are basically common to the set of protocols that are derived from it. Then it specifies the performance of the same, analyzing the novelties which are introduced at each stage in detail, with respect to AntHocNet. Once AntOR has been described, the chapter continues with the specification of the different protocols deriving from that, starting with the disjoint-link version (AntOR-DLR) as well as its variants AntOR-RDLR, AntOR-UDLR, AntOR-v2 and HACOR. Then, the version disjoint-node (AntOR-DNR) is analyzed as well as its parallel approaches PAntOR and PAntOR-MI. The chapter ends with a brief synthesis of the what has been shown.

6.1 Ant Optimized Routing (AntOR): Overview

AntOR is inspired in the AntHocNet algorithm, more specifically, in the specified version by the Ducatelle's Thesis [Duc07], inheriting its characteristics of hybrid (reactive and proactive), multipath and adaptive protocol. Like its predecessor, it presents the following phases:

- Routing setup: When starting data session, the source node, on demand, sends agents to discover the available routes to the destination.
- Data routing: the data is sent out through the nodes to the destination using the route information, being able to utilize the multi-hop technique, i.e. sending data through intermediate nodes that act as routers.
- Established path maintenance and exploration of new routes:information of existing routes is updated and it tries to discover new ones. This phase consists of two stages: pheromone diffusion and ant proactive sending.
- Management of link failures: These occur because a node is out of the reach of the

network or because it does not receive control messages which are responsible to inform a node of its closest neighbors.

Likewise, and similarly to its predecessor, with independence of the phase in which we are, each forward process (from the source node to the destination node) has its corresponding backward process.

In so far as the main differences of AntOR regarding AntHocNet, basically these consist in the introduction of the following elements/processes:

- Specification of Disjoint link or node routes.
- Separation between the pheromones in the diffusion process.
- Use of *distance* metric in path exploration.

These three characteristics influence especially in phases 1 and 3 of the algorithm, that is, in the setup phase and maintenance and exploration of new routes.

6.2 Ant Optimized Routing (AntOR): Data Structures

Like almost all of ACO routing protocols, AntOR requires two data structures: the Routing Table and the Neighbor Table. These have similar functionality to which other routing protocols have. Each of these are specified straight away.

6.2.1 Routing Table

Like all routing ACO algorithm for mobile ad hoc networks, the information related to routing is organized into the so-called routing tables. This data structure is present also in ACO routing protocols for wired networks as AntNet or routing classic protocols for mobile ad hoc networks as AODV.

These tables contain of the utilized information by the algorithm in its forwarding local decisions. The kind of information that it contains, as well as the way in which it is used and updated, depends solely on the characteristics of the algorithm. The routing table is in turn, a local database and a local model of the global state of the network.

This table consists of the following fields:

- *Regular pheromone* (τ_{ij}^{d}): It indicates the path where the data travels through. It is a heuristic value that contains an estimate of *goodness* to relay data packets along the route that goes from i to the destination d with next hop j. This value is expressed as the inverse of a time estimate or *cost* as was explained when introduced the equation 6.7. This cost is based on the metric used for the algorithm evaluation.
- *Virtual pheromone*(ω_{ij}^{d}): It indicates a path that can possibly be good. This virtual heuristic value has the mission of *auxiliary* value and is utilized as an alternative. It is created or updated in the pheromone diffusion process.
- *Número medio de saltos* (h_{ij}^{d}): It utilizes in the local route repair process to indicate how long the process needs to run correctly.

The *regular pheromone* and *average number of hops* values are related in the following way: when a route has a value from one, it also has one from the other. This is due to the fact that these two values are involved in the use of the backward ants in the reactive process, the proactive (exploration of new alternative routes) and the local route repair.

However this, the *virtual pheromone* value is created or updated independently, because it is utilized in the pheromone diffusion process.

Table 6.1 shows the structure of the routing table in AntOR. This structure stores the following information for each entry: reachable destination of the data session, next hop which the data are routed, value of *regular pheromone* and *virtual pheromone*, and the *average number of hops*.

Dynamically this table grows according to the calculation of reachable routes.

Table 6.1: Routing table (AntOR)

Values / Entries	**Destination**	**Next Hop**	**Regular Pheromone Value (τ)**	**Virtual Pheromone Value (ω)**	**Average Number of Hops (h)**
$Entry_1$	$Destination_1$	$Next\ Hop_1$	τ_1	ω_1	h_1
$Entry_2$	$Destination_2$	$Next\ Hop_2$	τ_2	ω_2	h_2
...	...	...	...	...	...
$Entry_i$	$Destination_i$	$Next\ Hop_i$	τ_i	ω_i	h_i
...	...	...	...	...	...

Figure 6.1 shows that intermediate node B has two routes associated with two data sessions with destinations G and D, respectively. The route with source and destination G (black line) and the route with source A and destination D (red line) have in common the intermediate node B. Table 6.2 represents its routing table.

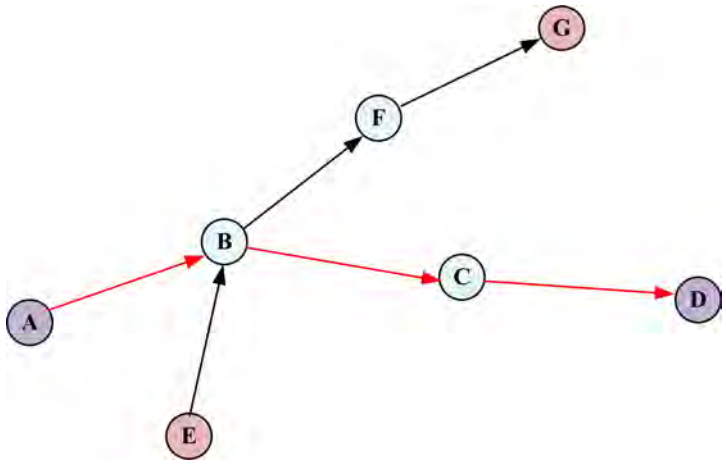


Figure 6.1: Updating scenario of routing table (AntOR)

This table 6.2 shows how the two route setup processes to nodes G and D imply that

there is as a regular pheromone τ value as a value h associated to the average number of hops. We also observe that it does not dispose of a virtual pheromone ω value because in this case there is no pheromone diffusion.

Table 6.2: Routing structure of node B (AntOR)

Route of Node B	**Destination**	**Next Hop**	τ	ω	**h**
Entry 1	G	F	τ_1	-	h_1
Entry 2	D	C	τ_2	-	h_1

6.2.2 Neighbor Table

This data structure contains the information that each node has of the one-hop neighbors with its corresponding *listening* time. Neighbors table maintained by the node i is a vector with an entry for each one of its neighbors. Each entry corresponds to the information that the node i has the presence of the neighbor node j, as well as a *time* value that indicates when was the last information was received from it, that is, that i received a message from j. This structure is utilized, as its name suggests, to indicate the presence of the neighbors and to detect possible link failures. Table 6.3 represents the generic table from the neighbors of AntOR. In this structure every local node has a list of one-hop neighbors with the following information: a neighbor identifier *Id* $Neigh_k$ and the last time value *Time* $Neigh_k$, associated with the notification message from vicinity (*Hello*), which neighboring node was sent.

Table 6.3: Neighbor table (AntOR)

Local node	Id $Neigh_1$	Id $Neigh_2$	...	Id $Neigh_k$	...	Id $Neigh_N$
	Time $Neigh_1$	Time $Neigh_2$		Time $Neigh_k$		Time $Neigh_N$

Figure 6.2 shows an example of a neighbor table. We can see that it corresponds to a network with 3 nodes that exchange messages with each other to update its neighbor tables.

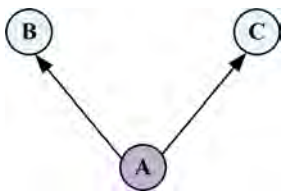


Figure 6.2: Scenario that represents the neighborhood of node A (AntOR)

Taking node A as a point of reference, after receiving the corresponding messages B and C, its neighbor table would be as shown in table 6.4. This table represents the identifier ID of each neighboring node from node A, as well as updated listening time.

Table 6.4: Neighbor structure of node A (AntOR)

Node A	Id Neighbor B	Time Neighbor B
	Id Neighbor C	Time Neighbor C

6.3 Ant Optimized Routing (AntOR): Functioning

6.3.1 Route Setup

Initially the nodes do not have routing information to send the data, but they have applications to start: traffic generator, *ftp*, *ping*, ..., the network interfaces, the protocol stack (IP, UDP/TCP and so on). The application generates data in the node, but with no available route, they cannot send them. Therefore, the node needs to send reactive agents (reactive ants) to discover the routes to the destination.

6.3.1.1 Reactive Forward Process

At the beginning of the route setup process, the node s, source of the session data, creates a *Reactive Forward Ant* (RFA). This ant is a control packet which aims to find a path from s to a given destination d. This ant goes from the source node to the destination node, being sent by s in broadcast mode.

The intermediate nodes that receive this ant, forward it in the route searching process until reaching the destination. This type of ants has a list P of visited nodes so that intermediate nodes are not repeated.

The forwarding mode of the RFA at the intermediate nodes may be unicast or broadcast, depending on if the current node has available routing information from to the destination d. In general, the RFAs are sent in broadcast mode because it aims to discover the first route. Unicast mode is utilized whenever the current node has information of a neighboring router that serves to relay the correspondent RFA to the next hop. In other words, a node has routing information whenever it is done the route setup, utilizing the first setup in broadcast mode for the sending of RFAs and in the subsequent ones (because link failures at source nodes) this mode or unicast, due to the remains of routes belonging to other previous setups.

Unicast forwarding is performed utilizing the equation 6.1 probabilistically, where τ_{in}^{d} is the regular pheromone value of the link that goes from node i to the next hop n in route to the destination d, N_i^d is the set of neighbors of node i with an available route to d and β_1 is an setting parameter influencing the exploratory behavior of ants.

$$P_{in}^{d} = \frac{(\tau_{in}^{d})^{\beta_1}}{\sum_{j \in N_i^d} (\tau_{ij}^{d})^{\beta_1}} \qquad \beta_1 \geq 1 \tag{6.1}$$

The value β_1 is determined experimentally. If we utilize a high value of β_1, the routes with a higher pheromone regular concentration are the candidates to relay the RFAs, obtaining the initial route quickly. If, on the contrary, we set to a lower value, routes tend to be chosen with similar probability.

More precisely, the route selecting process of the equation 6.1 is as follows:

When a node has the possibility of doing the hop to its neighboring nodes to get the destination d, it calculates the probabilities P_{in}^{d} of each of these neighbors n with the

regular pheromone value. According to this strategy, we do not choose the routes as a priority in which we are going to use them, but we select them as follows:

- It is generated with a random number *rand* with uniform probability between 0 and 1.
- The non-overlapping associated intervals to the calculated probabilities P_{in}^{d} are calculated above. These intervals are associated with each possible neighboring node at the time of selecting the candidate to transmit the message.
- Once *rand* is obtained, the associated route is chosen with the interval that corresponds with P_{in}^{d}. For this, the reactive ants are forwarded to the next hop n having as destination d.

Equacion 6.1 is based on a selection mechanism, widely utilized in genetic algorithms, called the *roulette selection*. This mechanism is also known as fitness proportionate selection. Where N is the number of existing individuals and f_i the fitness of the i-th individual, the associated probability to its selection is given by the following equation:

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_i} \tag{6.2}$$

Figure 6.3 shows an example of the selection of an individual. It verifies the condition:

$$\sum j = N_i^d (\tau_{ij}^d)^{\beta_1} = F$$

i.e., the sum of all regular pheromone values raised to parameter power is the total *fitness*. In this example there are 4 intervals labeled with letters, choosing B because the random number *rand* is included on it.

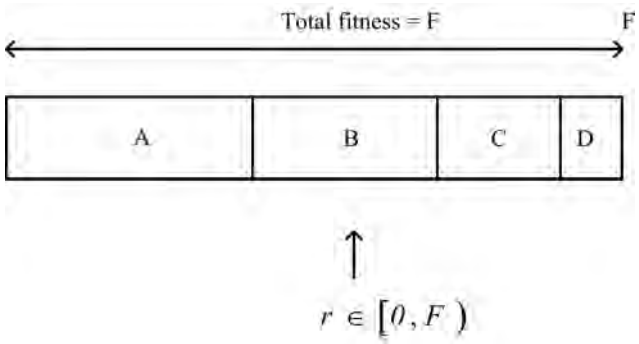


Figure 6.3: Example of the selection of an individual

On the other hand, as mentioned above, when the current node does not have routing information to a destination, the ant is always forwarded in broadcast mode. Due to these diffusion processes (including the initial broadcast at the node source s), a reactive forward ant may proliferate quickly. This generates different copies of the ant that follow different paths to the destination. AntOR limits the generated overhead, making sure that the nodes only forward the first copy of the ant that they receive, discarding the subsequent.

Figure 6.4 illustrates the first route setup process.

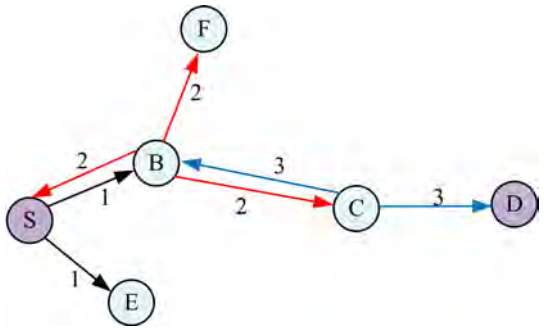


Figure 6.4: First route setup process (AntOR)

The arrows in this mentioned figure indicate the transmission range. The scheme is as follows: the node S initiates the first route setup to the destination node D by sending a RFA in broadcast mode. Then, nodes B and E receive the ant (process 1) to forward it later. Node B is located with two sending alternatives: unicast or broadcast. Upon being in the first route setup and not having any information in the routing table that indicate how to reach the destination D, the broadcast mode is utilized (process 2) to forward the RFA. The behavior of node E is similar to node B. To simplify, we do not represent the same in the figure. At this point, S, C and F receive the message from B. S discards it because it already has the information because it is the source node. For its part, C and F do the same process as B does, although, for reasons of convenience, the behavior of F is not indicated in the figure. Finally, B and D receive the message from C (process 3). B discards it because it has already processed it. D, on the other hand, it processes it since it is the destination node, sending the corresponding reactive backward ant to set the pheromone values between the destination D and source S.

6.3.1.2 Reactive Backward Process

Upon reaching the destination the RFA becomes a *Reactive Backward Ant* (RBA). The latter follows the list of visited nodes generated by RFA to return to the source node s. In this process, only the first copy of forward ant coming is chosen, discarding the remaining. In this way a unique route is set and, as mentioned previously, the overhead is reduced.

As discussed in section 4.3 of this Thesis, artificial ants are inspired by natural ants, but have some additional capabilities that improve their performance. So, while natural ants deposit pheromone coming and going, the artificial ants have an internal memory which store tour nodes information. This information is utilized by the backward ants in the return, reason why the return of the ant to the source is done in unicast mode. In this trip the backward ant is responsible for creating or updating a record in the routing table. This registry stores a *value* that represents the inversion of the cost in terms of estimated time to go a data packet from the destination node to the source through intermediate nodes.

Incrementally the backward ant calculates an estimate or *cost* c_i^d of time that would take a data packet to travel through that list P of nodes to the destination d from the node i, updating the routing tables.

The updating process from the routing table registry is as follows: the backward ant updates the number of hops h_{in}^d and the regular pheromone value τ_{in}^d from the routing table registry, being n the previous visited node, i the current node (which is currently being processed) and d the destination of the session.

Equation 6.3 summarizes the updating process of the number of hops h_{in}^d:

$$h_{in}^d \leftarrow \alpha h_{in}^d + (1-\alpha)h \qquad \alpha \in [0,1] \tag{6.3}$$

In this equation h is the number of hops that backward ant has traveled and α a regulation parameter that indicates how fast the formula to the new information is adapted. In experiments α has always been set to the usual value of 0.7.

The regular pheromone update process is as follows:

The estimate c_i^d commented previously is calculated according to equation 6.4, that is, it comes to be the sum of the time estimates that take to reach the next hop at each node of the list P:

$$c_i^d = \sum_{i=1}^{n-1} \hat{T}_{i \to i+1} \tag{6.4}$$

The value of the local estimate $\hat{T}_{i \to i+1}$ is defined as the product of two terms:

- The current number of packets in the queue which are ready to send at MAC layer plus one, this is:

$$Q_{mac}^i + 1$$

- The required average time to send a packet

$$\hat{T}_{mac}^i$$

with what $\hat{T}_{i \to i+1}$ is as shown in the equation 6.5:

$$\hat{T}_{i \to i+1} = (Q_{mac}^i + 1)\hat{T}_{mac}^i \tag{6.5}$$

If we consider the real time t_{mac}^i that takes a node to send a packet:

$$\hat{T}_{mac}^i \leftarrow \eta \hat{T}_{mac}^i + (1-\eta)t_{mac}^i \qquad \eta \in [0,1] \tag{6.6}$$

In the experiments η has also set to 0.7. With this parameter we want to indicate that $\hat{T}_{mac}^i$ has more priority than t_{mac}^i, specifically 70%. The value t_{mac}^i at each hop is estimated in AntOR as the time difference between the sending and receiving of the backward ant.

Finally, the updating of regular pheromone value is calculated as shown in equation 6.7:

$$\tau_{ij}^d \leftarrow \gamma \tau_{ij}^d + (1-\gamma)(c_i^d)^{-1} \qquad \gamma \in [0,1] \tag{6.7}$$

Using the previous equation the value of a registry $\tau_{ij}^d$of the routing table from node i is updated, j being the next hop, d the destination that we want to reach and γ a setting parameter set to 0.7 in the performed experiments.

In particular case, there is virtual pheromone in the link / arc that we want to update (as consequence of which the diffusion process is completed before the *route setup process*), that is, if the node i that has a route to the destination d using next hop j already has virtual pheromone, the update of the pheromone regular in the *route setup process* described by the equation 6.7 is, for the process of regular and virtual pheromone separation, as follows:

$$\begin{aligned} Regular_{final} &= F(Regular_{new}, time) \\ Virtual_{final} &= 0 \end{aligned} \tag{6.8}$$

being

$$Regular_{final} = F(Regular_{new}, time)$$

a simplified representation from the equation 6.7.

In other words, when we get a new regular pheromone value, the pheromone virtual value is set to 0. We give priority, therefore, to regular pheromone regulate against the virtual, since data are routed only along routes with regular pheromone value. Thus, we do not originate any conflict in the creation and maintenance (updating) of the routes being the algorithm optimized with regard to its capacity (internal memory) due to route table only has an entry by destination and next hop. This entry can contain its corresponding field of pheromone regular or virtual, but not both.

Figure 6.5 shows an example of the backward process of updating the pheromone and the average number of hops in the routing table. The RBA goes from the destination node D to the source node S as shown by the arrows. On its way back, values from the routing table are created or updated according to equation 6.7. The process is as follows: D sends a unicast message to RBA to node C. In the instant that C receives it, creates the corresponding registry entry in the routing table.

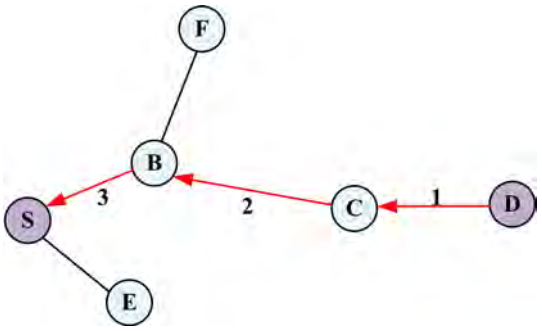


Figure 6.5: Example of pheromone settlement (AntOR)

The contained information in the register is as follows:

- Destination
- Next hop
- Regular pheromone value
- Estimate value of number of hops

The node B and S perform the same operation as C.

Figure 6.6 shows a summary scheme of the functioning of route setup phase. It is important to note that the creation of multiple routes to the same destination is not carried out at this phase; it takes place in the proactive route exploration process that will be seen later, specifically in the paragraph .

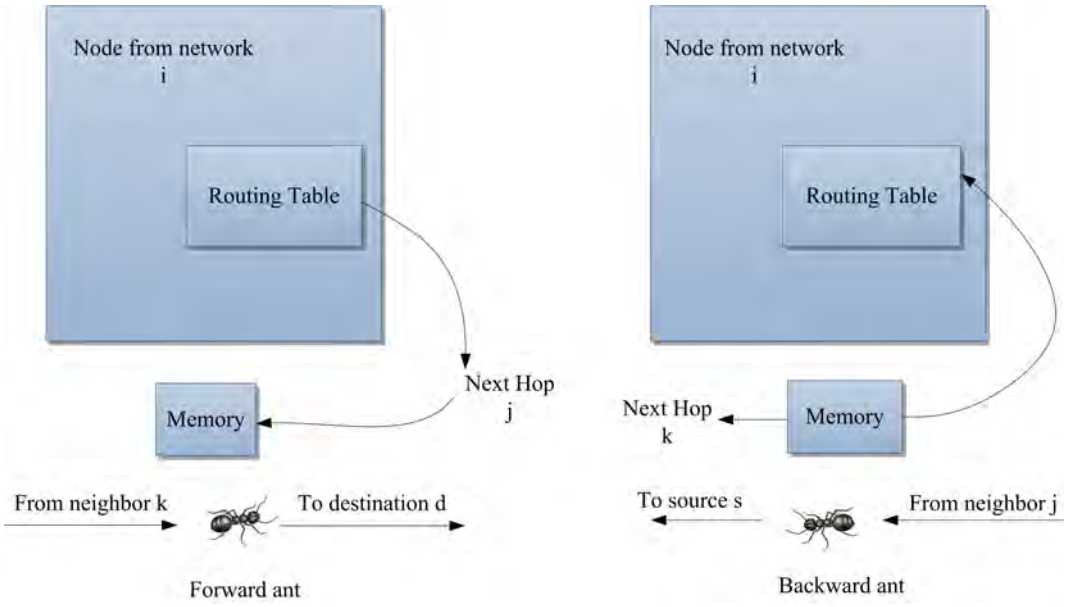


Figure 6.6: Functioning of route setup process (AntOR)

6.3.2 Data Stochastic Routing

The first route setup creates a unique path between source and destination, as shown in the routing table. Other route discoveries and the route exploration which is explained in the following section originate multiple paths between source and destination. This being carried out means the data can be forwarded in mode multi-hop according to a probabilistic technique based on the routing tables. The strategy consists of making the data load expand through load balancing. This is important in mobile ad hoc networks because of the wireless channel bandwidth is very limited.

The data routing is given by the following equation:

$$P_{in}^{d} = \frac{(\tau_{in}^{d})^{\beta_2}}{\sum_{j \in N_i^d} (\tau_{ij}^{d})^{\beta_2}} \qquad \beta_2 \geq 1 \tag{6.9}$$

Equation 6.9 is similar to 6.1. The difference is in the exponential parameters β_1 and β_2.

6.3.3 Established Path Maintenance and Exploration of New Routes

As its name suggests, this phase consists of a proactive process of established route maintenance and exploration of new routes, which updates and expands the available routing information. This allows us to build multiple routes that serve as support for the created initial route in the reactive route setup process. This proactive process contemplates two subprocesses: *pheromone diffusion* and *proactive ant sending*. This phase of AntOR differs from the similar AntHocNet in the separation of pheromones in the diffusion process, the disjoint ability and the use of the *distance* metric, differences that affect directly the two sub-processes of this phase.

6.3.3.1 Pheromone Diffusion

In AntOR a route to a destination cannot have a regular pheromone and virtual pheromone value simultaneously. This restriction or characteristic of AntOR is not referred to in AntHocNet and is called property of separation between the pheromones.

Pheromone diffusion aims to expand the information available from pheromone in the network by sending updating periodic messages and the bootstrapping technique to know reachable destinations in the network. This process is similar to the pheromone diffusion in the nature. The *Hello* messages play an important role: every certain interval of time t the nodes send messages of this type in broadcast mode. The experiments t was set equals 1 second. These messages are also used to know the 1-hop neighbors and to detect link failures. At the same time these messages serve to spread the necessary pheromone in the bootstrapping process.

These generation *Hello* message is as follows:

A node i chooses a maximum number k of destinations by consulting the information in its routing table. When there are more available destinations, these are selected randomly. Experimentally the value of k is set to 10 (good results were obtained for this value and not just overload). For each destination d, *Hello* message has information of the best pheromone value v_i^d that the node i has the destination d. This is calculated by taking into account all the possible regular pheromone values τ_{ij}^d and virtual pheromone values ω_{ij}^d associated with the destination d. In addition to including it, a *flag* indicates if the chosen value corresponds to a regular or virtual pheromone value.

Once created the *Hello* message is sent, as described earlier, in broadcast mode. All the neighbors of node, which sends this message *Hello*, receive a copy. Thus, a neighboring node j, upon receiving this message, estimated a new value that indicates how *good* the route is from this node j to the sender i which has a reachable destination d shown in the destination list of the *Hello* message. This estimation is made by combining (bootstrapping) the pheromone value v_i^d from *Hello* message with the local estimation or *cost* c_j^i of the hop j to i, i.e., the link between the node j and the node i; c_j^i corresponds here to $\hat{T}_{i \to i+1}$ from equation 6.4.

Equation 6.10 summarizes bootstrapping process:

$$k_{ji}^d = ((v_i^d)^{-1} + c_j^i)^{-1} \tag{6.10}$$

Being k_{ji}^d the bootstrapped value obtained in this process. Thanks to the use of this technique, the overhead is low, because the needed unique is to send the value v_i^d from node i to j. This information is included in the Hello message which is sent in broadcast and that when it is received by a node it never is forwarded (it would increase overload then).

This bootstrapping process is repeated constantly when the simulation starts with the sending of *Hello* messages in an asynchronous way by each node in the network. Although this process has low overhead it may have reliability problems. The value obtained by bootstrapping only is correct when it is the value v_i^d contained in the *Hello* message. This is especially problematic in highly dynamic environments where routing information is not updated quickly and, especially, if included value in the *Hello* message corresponds to virtual pheromone. For these problematic reasons, it should add the fact that the bootstrapping process is relatively slow, because the sending of *Hello* messages are carried out every certain interval time (in order to maintain its efficiency).

For the above considerations, AntOR has the premise that the bootstrapped pheromone value k_{ji}^d obtained in the equation 6.10, is hardly reliable. This feature directly affects the

updating of the routing table when we use this value k_{ji}^d and in the separation of regular and virtual pheromone values. Generally, the virtual pheromone value is updated with the new bootstrapped pheromone value. On the other hand, only regular pheromone is updated by this bootstrapped pheromone value (k_{ji}^d) when the following conditions occur simultaneously:

a) The node j which receives the corresponding *Hello* message has a non-zero regular pheromone value.

b) This *Hello* message also contains a value v_i^d corresponding to regular pheromone.

In addition, AntOR applies the following premise:

If node j which has a route to the destination d already has regular pheromone and it gets virtual pheromone contained in the *Hello* message during pheromone *diffusion process*, then the virtual value is not updated at node j, since it cannot simultaneously have non-zero values in both pheromones. The value of final virtual pheromone is, therefore, zero, as it picks up the equation 6.11.

$$\begin{aligned} Regular_{final} &= Regular_{old} \\ Virtual_{final} &= 0 \end{aligned} \tag{6.11}$$

As described earlier in subsection 6.2.1, another advantage obtained by applying this restriction of separation between pheromones is that virtual pheromone behaves as a backup value (alternative) in the exploration of new routes, while data are only routed by regular pheromone paths.

In addition, it should be noted that the ants from pheromone diffusion also serve to detect broken links. So, when there is a failure link, nodes can update their routing tables.

To understand this better, it is explained in this paragraph, Figure 6.7 shows an example of selection of the best value of pheromone. The red arrows indicate regular pheromone links and the black color ones a link of virtual pheromone. The current node A to create its corresponding Hello message by selecting an reachable destination (D in this case) and choosing the best value of pheromone between the available values ω_{AC}^D and τ_{AB}^D.

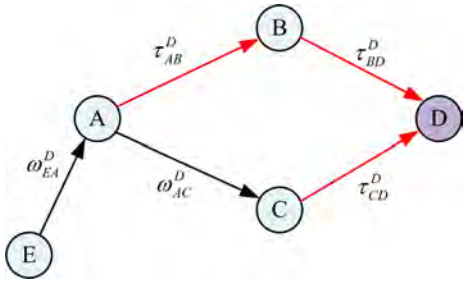


Figure 6.7: Example of selection of the best pheromone value (AntOR)

We suppose that it is chosen the value of virtual pheromone ω_{AC}^D as value v_A^D. Then the sending of the Hello message in broadcast mode is done. Now suppose that the node E receives the Hello message sent by A. The new bootstrapped value of the link between E and reachable destination D is obtained by applying the following equation:

$$k_{EA}^{D} = ((v_{A}^{D})^{-1} + c_{E}^{A})^{-1} \tag{6.12}$$

According to the equation 6.12 the node E gets a new value of pheromone by combining the information of the *Hello* message with the estimated *cost* of the link between E and its neighbor A from which it received the message. The inversions of this equation are necessary, firstly to convert a value of pheromone to a value compatible with the *cost* c_{E}^{A} and thus be able to make the sum, and secondly to return converting the total sum at value of pheromone. This final value k_{EA}^{D} can be used to update ω_{EA}^{D} or τ_{EA}^{D}. As the premise of the equation 6.11 is not satisfied, ω_{EA}^{D} is updated.

6.3.3.2 Ant Proactive Sending

This sub-process involves the exploration of new routes by sending proactive ants. This sub-process takes into account the disjoint link/node property, so appropriate, firstly, to discuss the general characteristics of this type of routes, then to explain the functioning of the exploration of new routes basing on the *distance* metric.

6.3.3.3 Disjoint Link / Node Routes

Disjoint link/node routes are those that do not share links/nodes, respectively. Disjoint link routes are less restrictive and easier to calculate, but they have a lower level of fault tolerance than disjoint node.

The property is satisfied that every disjoint-node is also a disjoint link, but not vice versa.

Both types of disjoint routes have the following advantages:

- A failure in one node only affects a path, not the entire network.
- Load balancing is better because there are not repeated routes on the disjoint property.

However, the use of such routes does also have its disadvantages:

- More resources are needed because they do not share links and nodes.
- These routes are more difficult to detect, because we limit the nodes that can be explored.

There are two modalities of creating of disjoint route, so-called A and B.

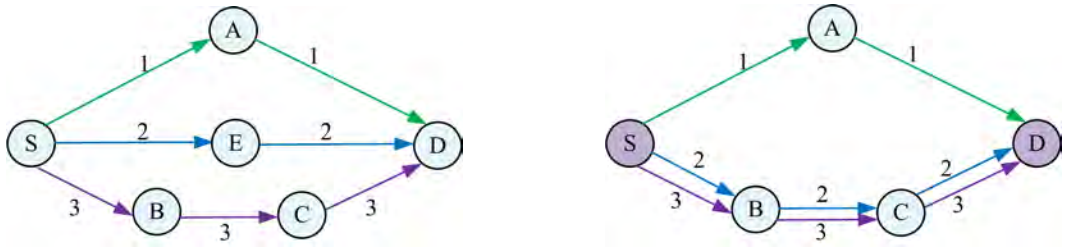
In the first (modality A), once created the main route for route setup and when necessary to discover new ones, to take into account the premise that these routes are disjoint with respect to the main route, so these alternatives can be repeated. This makes it to be a mode less restrictive, and it can update more frequently the routing tables, even though it carries a higher overhead for sending more proactive agents.

The second (modality B) creates a main route and other alternatives routes all disjoint among themselves. This modality is more restrictive but causes a minor overload.

The modality B has a better tolerance to link failure than the modality A because it enables a greater number of disjoint paths, presenting the drawback about alternative routes may not be updated at this moment.

After various experiments is opted in the simulations by the modality B due to the exposed considerations: lower overhead and greater range of disjoint routes.

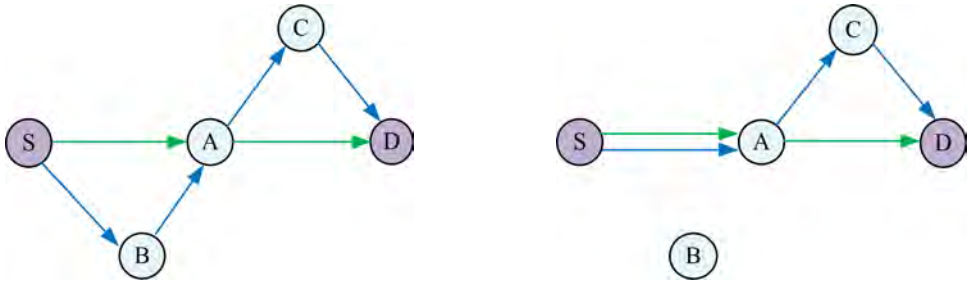
Figure 6.8 presents the comparison of the two modalities of disjoint route calculation. The main route is labeled with the number 1 and the alternative routes which are calculated with the numbers 2 and 3. It is observed how the modality A, an already created alternative route (number 2) is recalculated by second time, at the same time that the pheromone tables from the nodes that constitute it are updated (number 3). This keeps the updated routes at the expense of high overhead, since more routes exploration proactive ants are sent. On the other hand, the network associated to the modality B is formed by disjoint routes among themselves, resulting in lower overhead, better tolerance to link failure and lower frequency of route update.



(a) Modality A (b) Modality B

Figure 6.8: Comparison of the two modalities (AntOR)

Figure 6.9 illustrates the concept of disjoint link routes graphically. Figure 6.9(a) represents a disjoint link route, where node A does not share links. It is shown that the nodes S and A share only a single link (S-A). By contrast, Figure 6.9(b) represents the non-disjoint link route version because the nodes S and A share two overlapping links of the same data session.



(a) (b)

Figure 6.9: Disjoint link routes a) versus non-disjoint b) (AntOR)

Figure 6.10 illustrates the concept of disjoint node routes. Figure 6.10(a) we can observe two routes: the main (color green) and the alternative (color blue). These routes do not share nodes considering disjoint node. By contrast, Figure 6.10(b) shows the case of non-disjoint node routes since at the node A, there are two overlapping routes: the main and the alternative.

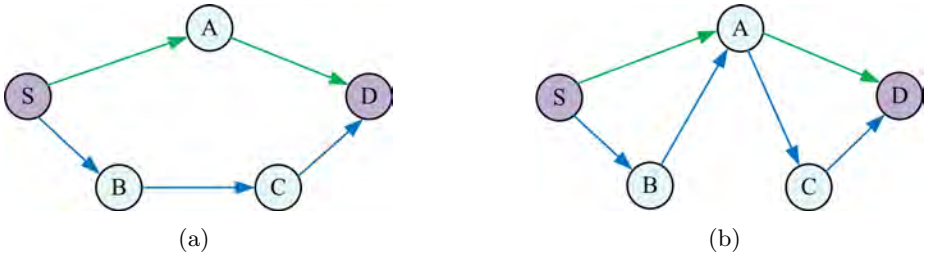


Figure 6.10: Disjoint node routes a) versus non-disjoint b) (AntOR).

Figures 6.11 and 6.12 we illustrate a comparative graph between link and node disjoint routes. In these, a scenario formed by 6 nodes is presented, in which the node source A has two created routes (green and blue) to the destination F.

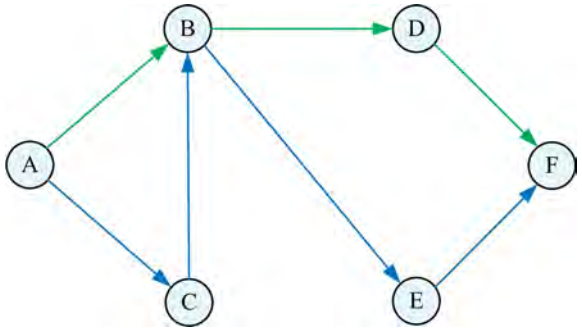


Figure 6.11: Scenario: Disjoint link route (AntOR)

Figure 6.11 corresponds to an example of disjoint link route and Figure 6.12 to another of disjoint node route. On this aforementioned scenario two link/node failures occur:

a) Fail the sending of message between the link (B-D)

b) Fail the node B (out of the range of transmission or disabled)

In case a) the scenario from Figure 6.11 has an alternative route formed by the link (B-E) for sending the information. However, in the scenario from Figure 6.12 has to do a process of link failure neutralization of because we observe that the intermediate nodes, in the disjoint node routes, cannot have alternative paths.

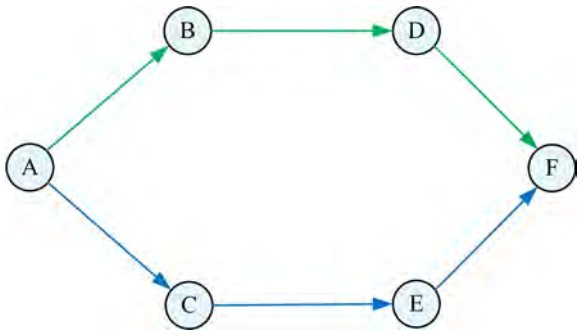


Figure 6.12: Scenario: Disjoint node route (AntOR)

In case b) the scenario from Figure 6.12 offers a better tolerance to the failure than the one of the Figure 6.11 because it employs the second alternative (blue) to send the message to node C (via link A-C). On the other hand, the scenario from Figure 6.11 shows the fact that if the node B disappears or disables, two routes (green and blue) are broken, making it impossible to communicate with the destination F until another route setup process.

This simple example explains how disjoint node routes are more restrictive, more tolerant to failures, more difficult to calculate, and in some particular scenarios, may be worse than the disjoint link routes. Also, with this example, it is clear that all disjoint node route is also disjoint link, but not vice versa.

6.3.3.4 Functioning

Exploration consists of a process to discover new routes that serve as alternatives for the sending of data packets. Diffusion process discussed earlier is essential for the correct functioning of the exploration. The source node of a session node starts this exploration process at the time that the destination node receives the first data packet from a new session. This process is maintained while the session is active. Initially it generates the corresponding *Proactive Forward Ant* (PFA) for subsequent sendings. These ants are never sent in broadcast mode, since they only go by paths that have marked the route, either by regular pheromone, like with virtual pheromone.

In AntHocNet [Duc07], for reasons of efficiency, only a proactive forward ant is sent if the best value of virtual pheromone is higher (at least by 10%) than the corresponding regular pheromone. This feature does not apply in AntOR due to the disjoint property and the *distance* metric limit the sending of proactive forward ants, which would lead to a more restrictive hypothesis. In AntHocNet [Duc07] the equation of route exploration is as follows:

$$P_{in}^{d} = \frac{[max(\tau_{in}^{d}, \omega_{in}^{d})]^{\beta_3}}{\sum_{j \in N_i^d} [max(\tau_{ij}^{d}, \omega_{ij}^{d})]^{\beta_3}} \qquad \beta_3 \geq 1 \tag{6.13}$$

In AntOR the equation of exploration is as follows:

$$P_{in}^{d} = \frac{(\psi_{in}^{d})^{\beta_3}}{\sum_{j \in N_i^d} (\psi_{ij}^{d})^{\beta_3}} \qquad \psi \in \begin{cases} \omega & virtual \\ \tau & regular \end{cases} \tag{6.14}$$

where ψ is a regular or virtual pheromone value and β_3 a setting parameter relative to the influence of the pheromone concentration (with similar functionality to β_1 and β_2).

It is worth mentioning that AntOR uses the *distance* metric, circumstance that does not occur in AntHocNet [Duc07]. Thus, it is considered the number of hop from the best routes found.In this way, a proactive ant is controlled and cannot go more nodes than those set by the so-called hop limit, which is set according to the best routes (those with less distance in number of hops) calculated above. The reason for choosing this metric (and not others, such as the delay, for example) is that it is considered stable, since it does not influence the interference caused by other devices.

The PFAs arrive to their destinations becoming *Proactive Backward Ant* (PBA). These latest have a functionality of updating the routing tables analogous to the commented in RBAs from paragraph 6.3.1.2.

Figure 6.13 shows an operation example of this routes exploration stage in regard to the separation of the pheromones.

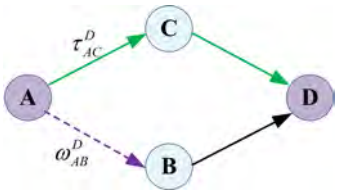


Figure 6.13: Example of route exploration (AntOR)

In the example the route alternative 1 (continuous green line) has only regular pheromone and route alternative 2 (discontinuous purple line) only virtual pheromone. The choice probability of alternative route 1 is given by the following equation:

$$P^D_{AC} = \frac{(\tau^D_{AC})^{\beta_3}}{(\tau^D_{AC})^{\beta_3} + (\omega^D_{AB})^{\beta_3}} \tag{6.15}$$

Figure 6.14 illustrates an example of the use of the *distance* metric. It is a scenario about disjoint node route. Initially, it creates the main route, represented in green and labeled with M. For sending the corresponding proactive ant in the exploration process of new routes one of the two possible alternatives is chosen: alternative route A_1 or alternative route A_2. This choice is based on the cost (distance) that supposes to reach the source node S to destination D. In this example the most favorable alternative route (route candidate) to transmit (to send the PFA) is A_1 with next hop C. We can check how the alternative has 2 hops against 4 hops of the alternative A_2 and to 3 of main route M.

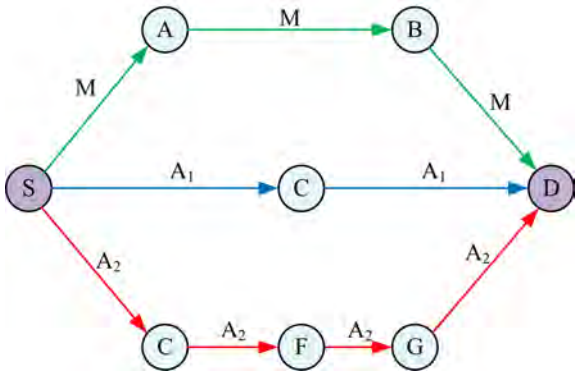


Figure 6.14: Exploration scheme using the *distance* metric (AntOR)

6.3.4 Management of Link Failures

Nodes can detect link failure in a unicast transmission or when a Hello message is expected and is not received. When a link fails, the node can lose the route to one or more destinations. An example of a link failure occurs when a neighbor moves beyond the transmission range. Link failures consider two kinds of problems:

- if the node has other alternatives to the destination or if the route to the destination is lost, because it has not been used regularly, it has to be notified with a link failure message. Thus, the node updates its routing table and it sends a failure notification ant in broadcast mode. This ant contains a list of the destinations that lost the way: new estimated End-to-End delay and the number of hops to this destination. All

its neighbors receive the notification and they update their pheromone table using the new estimates. On the other hand, if the neighbors lose their best or their only route to a destination due to the failure, they generate and send an failure ant in broadcast mode, until all the nodes from different ways have received notification of the new situation.

- If the route to a destination that is regularly used by the data is lost, and is the only alternative of the node, the loss is especially important and the node tries to repair the path locally. In AntOR, the node only repairs the path if it discovers that the loss link is due to the failure of a data packet transmission.

Figure 6.15 shows a neutralization process scheme of the link failures in AntOR.

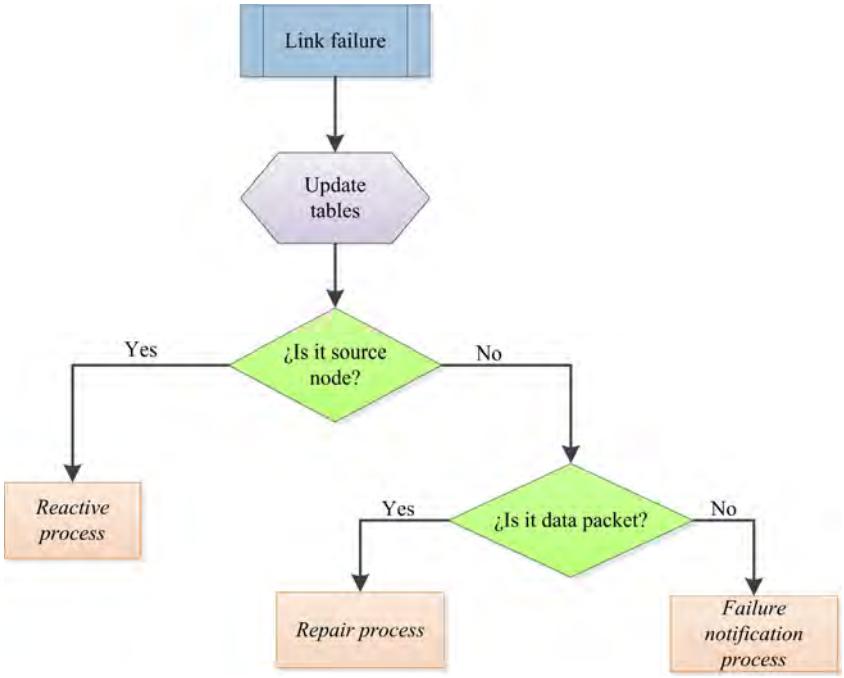


Figure 6.15: Link failure management (AntOR)

In AntOR-DLR, the first task that occurs when there is a failure node is that the node detecting the failure removes it from its neighbors table. Then, the routing table with the new pheromone information is updated. Finally, it is responsible for neutralizing the failure taking into account the following factors:

a) If there is not a route at the source, a reactive forward ant is sent.

b) If there is no route at an intermediate node and it is dealt with a data packet that had been forwarded when the failure occurred, a route repair forward ant is sent. If there is no reply from the corresponding repair backward ant in a certain time period a link failure notification message is sent in broadcast mode, reporting the unreachable destination.

c) When there is a link failure, due to the fact that the corresponding consecutive Hello message has not been received in a determined time while or because a unicast control message is lost at some intermediate nodes, a link failure notification message is created informing it about unreachable destinations and sending in broadcast mode.

One of neutralization mechanisms presented in Figure 6.15 is the route repair process which is very similar to a route setup: the node sends a *Route Repair Forward Ant* (RRFA) in broadcast mode and the intermediate nodes forward this ant in the same way, but with a maximum limit of attempts. However, if there is route information available at intermediate nodes, the sending through them is performed in unicast mode, applying the equation 6.1.

In this route repair process is needed the information of the number of hops h_{ij}^{d} that is in the routing table, since the node that starts the repair process expected to get a *Route Repair Backward Ant* (RRBA) some time:

$$T_{Wait} = 2t_{hop}h_{ij}^{d} \tag{6.16}$$

Equation 6.16 represents an estimate of the time that it takes to go and return from node i to node destination d. t_{hop} is set, the fixed value of delay per hop, to 50 ms.

If the corresponding RRBA is not received, the process is terminated, by not to repair the route in the established time by the equation 6.16. Consequently, the node that detects the failure discards the data packet previously enqueued (because it has not been fixed correctly the route according to the timer), generating an failure notification ant that announces the new situation.

Figure 6.16 illustrates a route repair process.

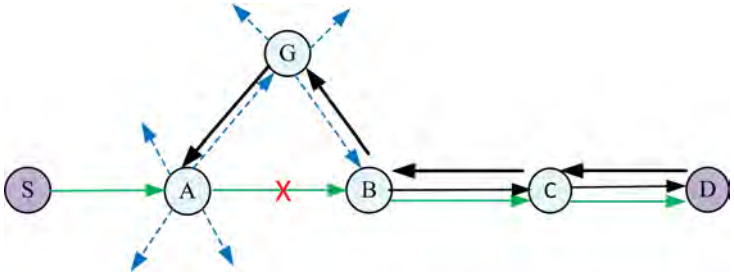


Figure 6.16: Example of local route repair (AntOR)

In this example the route from session data between the node S source and destination D (green color line and path S-A-B-C-D) is broken as a consequence of a link failure between nodes A and B. node A that detects the failure tries to repair it sending an RRFA to the destination D in broadcast mode (the route marked with a discontinuous line in blue color). G receives a copy of this ant and spreads it out. B receives the RRFA then it sends in unicast mode, because there are (black color) route between this node and the destination D which is the part of the original route which is valid. Finally, destination node D sends a RRBA to the local node A (path D-C- B-G-A). This ant, similar to reactive process, is responsible for updating the routing tables from the visited nodes in the return route.

6.3.5 Summary

Figures 6.17 to 6.19 show a complete example of the functioning of AntOR. The scenario of the Figure 6.17 shows that there is a single data session formed by the pair (A, D); in other words, A sends the information to the destination D.

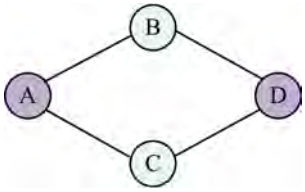


Figure 6.17: Example of marking and route settlement (AntOR)

It is referred by initiating the data session the fact that you want to send data from source S to destination D. With the *Hello* messages, routes which are independent from the data sessions are created between pairs of neighbors who are one hop. The example shows 8 routes: A-C, C-A, A-B, B-A, B-D, D-B, C-D, D-C.

Figure 6.18 illustrates the process of route marking and setting for the calculation of τ and h. This is done as seen in paragraph 6.3.1.2. In this example of virtual pheromone ω value is 0 for all routes.

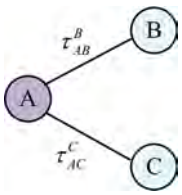


Figure 6.18: Scenario example of protocol functioning (AntOR)

The routes tables (from the 6.5 to the 6.8) are as follows:

Table 6.5: Routes for node A (AntOR)

Routes for Node A	**Destination**	**Next Hop**	τ	ω	**h**
Neighbors	B	B	0.3	0	0.3
	C	C	0.3	0	0.3

Table 6.6: Routes for node D (AntOR)

Routes for Node D	**Destination**	**Next Hop**	τ	ω	**h**
Neighbors	B	B	0.3	0	0.3
	C	C	0.3	0	0.3

Table 6.7: Routes for node B (AntOR)

Routes for node B	**Destination**	**Next Hop**	τ	ω	**h**
Neighbors	A	A	0.3	0	0.3
	D	D	0.3	0	0.3

Table 6.8: Routes for node C (AntOR)

Routes for Node C	**Destination**	**Next Hop**	τ	ω	**h**
Neighbors	A	A	0.3	0	0.3
	D	D	0.3	0	0.3

Another process that is performed regardless of the data session is the *diffusion process*. Suppose the following situation: node A has no route to D via C yet, so node C spreads out the information of its destination D to node A. Node C informs node A about the best route to the destination node D; in this case just one. If there were other alternatives, it would choose the best regular or virtual value to destination D.

Figure 6.19 shows the pheromone diffusion process where discontinuous line represents the virtual pheromone whose value (0.3) is based on the best destination from C to D.

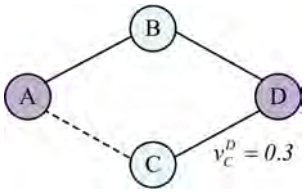


Figure 6.19: Scheme of routing diffusion (AntOR)

Node A, which receives information from C, applies the equation 6.10, staying the new routing table at this node, as shown in Table 6.9. If the event for which A has a regular value exists, the virtual value does not update in node A. This *routing diffusion* process is repeated constantly every time.

Table 6.9: Routes for node A in diffusion process (AntOR)

Routes for Node A	**Destination**	**Next Hop**	τ	ω	**h**
Neighbors	B	B	0.3	0	0.3
	C	C	0.3	0	0.3
	D	C	0	0.23	0

Finally it has been believed that it is convenient to note a general scheme of functioning of AntOR. Figure 6.20 shows such a scheme.

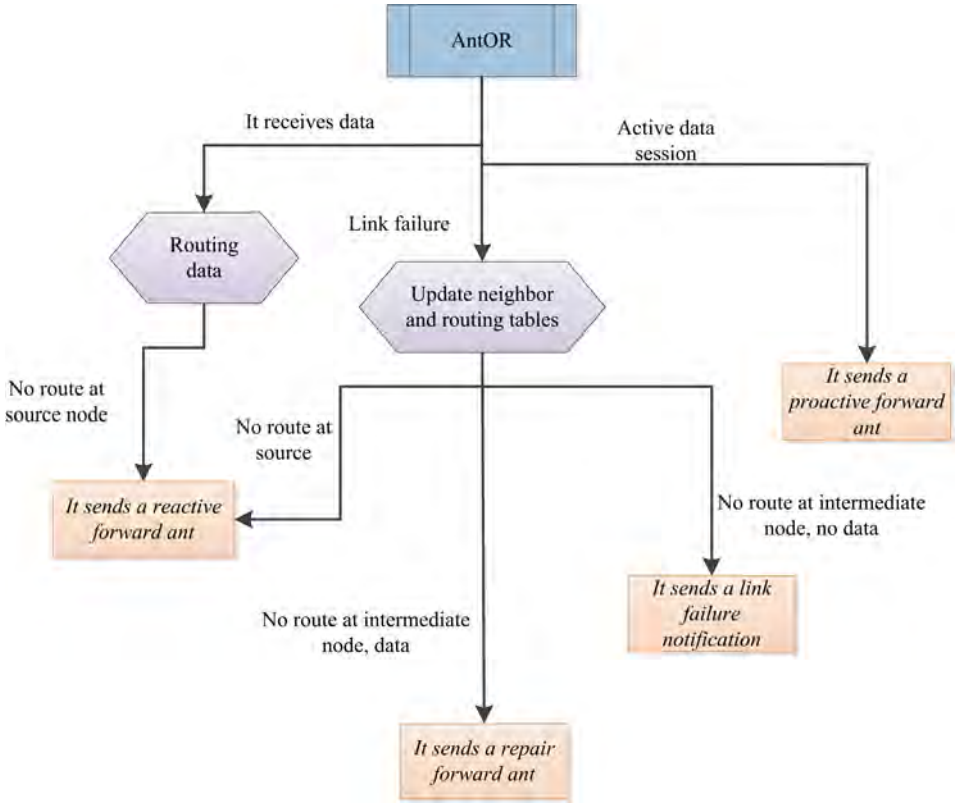


Figure 6.20: Functioning scheme of AntOR

6.4 AntOR - Disjoint Link Route (AntOR-DLR)

As its name suggests, AntOR-DLR is derived from the basis protocol AntOR with the only restriction that its specification takes into account only routes that do not share links.

Table 6.10 shows the routing table of AntOR-DLR. As we can see, the routing table adds regarding to AntOR an additional field called Disjoint Session.

Figure 6.21 shows a scheme of how link disjoint routes are constituted. The basic idea to find and to represent disjoint link routes consists of putting a *mark* on each disjoint link with a label indicating what the source of the data session is. This *mark* is indicated in the field *Disjoint Session* in the routing table discussed earlier.

Table 6.10: Routing table (AntOR-DLR)

Values / Entries	**Destination**	**Next Hop**	**Regular Pheromone Value (τ)**	**Virtual Pheromone Value (ω)**	**Average Number of Hops (h)**	**Disjoint Session (s)**
$Entry_1$	$Destination_1$	$Next\ Hop_1$	τ_1	ω_1	h_1	s_1
$Entry_2$	$Destination_2$	$Next\ Hop_2$	τ_2	ω_2	h_2	s_2
...	...	...	...	...	...	...
$Entry_i$	$Destination_i$	$Next\ Hop_i$	τ_i	ω_i	h_i	s_i
...	...	...	...	...	...	...

In Figure 6.21 shows a network consisting of 5 nodes and two disjoint routes (the main route of green color and the alternative with red color and a discontinuous line). In both routes are marked the links in the field *Disjoint Session* with the source of the data session. For example, node A has two entries in its routing tables for the links (A, C) and (A, B) with, among others, the following information: *destination* and *next hop* and *disjoint session*. *Entry 1* has the combination (D, C, A) and *entry 2* the combination (D, B, A).

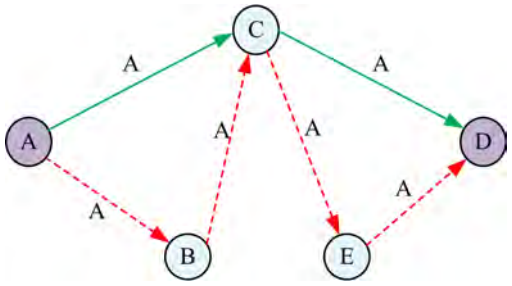


Figure 6.21: Representative scheme of disjoint link routes (AntOR-DLR)

Figure 6.22 shows the flowchart of the procedure of calculation of disjoint link routes. As we can observe, the procedure is as follows: It is consulted the *disjoint session* field in the routing table to verify if the link is already disjoint or not. For this end, we checks if the link *Link* is associated with the source of the data session. In negative case, it sends the corresponding proactive forward ant to the previously calculated next hop. Upon receiving this proactive ant the process is repeated in the intermediate nodes.

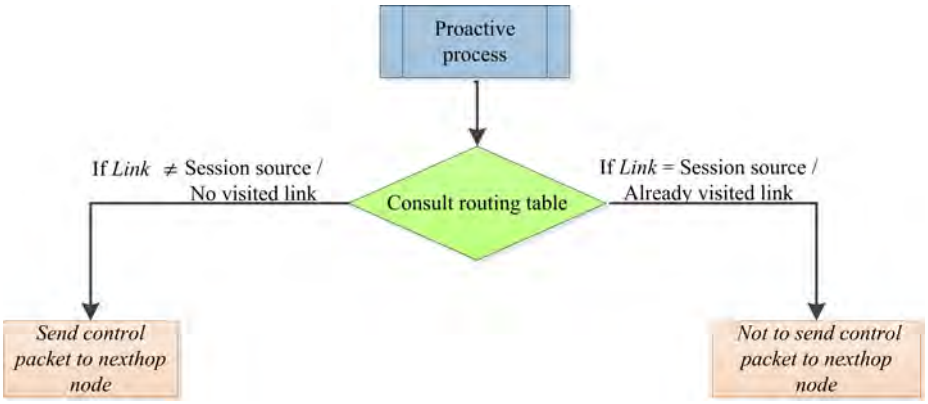


Figure 6.22: Flowchart of disjoint link routes (AntOR-DLR)

Algorithm 6.1 represents the pseudocode of the calculation process of disjoint link routes.

```
1 while Proactive Process do
2     {src,dst} = GetSessionInfo(msg);
3     {nexthop} = GetNextHop(dst);
4     {link} = GetLink(dst,nexthop);
5     if CheckDisjointLink(link) = FALSE then
6         Send(nexthop,msg);
7     end
8 end
```

Algorithm 6.1: Calculation of disjoint link routes (AntOR-DLR)

Line 1 shows a proactive process that is represented with a loop to indicate that it is performed after the start of the data session continuously. In lines 2 and 3 we get information from the data session (source and destination) and the *next hop* to be associated with the *destination*, respectively. Line 4 gets the link *Link* looking for the routing table from local node. In line 5 the method *CheckDisjointLink* checks the disjoint link property, i.e. if this link *Link* calculated above coincides with the source of the data session indicated in the PFA. If not (there is not a disjoint link route), a proactive ant (message *msg*) is sent to the next previously calculated hop (*nexthop*) .

Figure 6.23 shows an example of disjoint route calculation. In the above example there is only a data session formed by the source node S and the destination node D. First, we estimate disjoint route S_1 marking each link disjoint with the origin of the session S. After alternative route S_2 is calculated taking into account it is not to repeat links already belonging to the source S. The characteristic of being able to share nodes makes it possible for us to visit intermediate nodes for alternative routes, but not the links that belong to other routes.

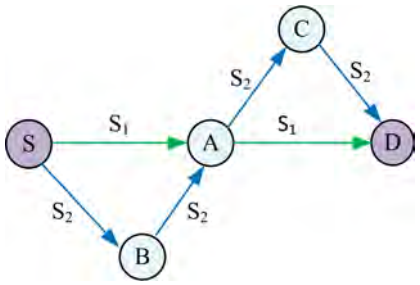


Figure 6.23: Example I: One data session (AntOR-DLR)

When we work with multiple data sessions, several pairs of source/destination, disjoint property is also performed, because the route marked is unique for each disjoint route belonging to each data session. Figure 6.24 shows an example in which belonging disjoint routes are overlapped belonging to different data sessions. It is observed that there are two data sessions formed by pairs (B-E) and (A-D), proving that, although there is overlap, for being different data session, the protocol behavior is not altered.

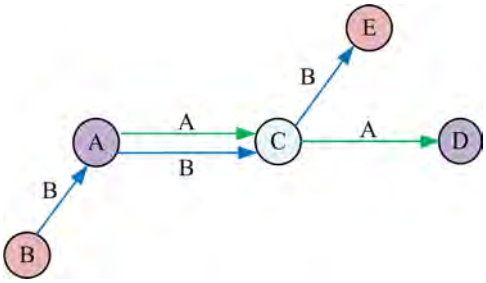


Figure 6.24: Example II: Two data sessions (AntOR-DLR)

6.5 AntOR - Disjoint Node Route (AntOR-DNR)

As its name suggests, AntOR-DNR is derived from the basis protocol AntOR with the only restriction being in its specification takes into account only routes that do not share nodes.

Like AntOR-DLR, the routing table of AntOR-DNR adds an additional field so-called Disjoint Session with respect to AntOR.

The main difference between AntOR-DNR and AntOR-DLR consists in the way of calculating the routes in the exploratory process: in the disjoint node routes is the node responsible for detecting the disjoint property, while disjoint link routes is the own link.

Figure 6.25 shows a diagram of how disjoint node routes are constituted. We observe a network formed by 5 nodes, a main route of green color and a possible disjoint alternative route with red color and a dashed line.

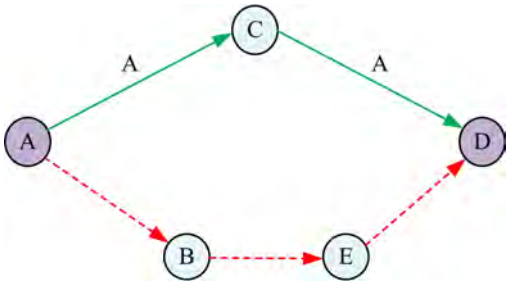


Figure 6.25: Representative scheme of disjoint node routes (AntOR-DNR)

Figure 6.26 contains the flowchart of the functioning of AntOR-DNR. The protocol works as follows. Initially, it is sent by the corresponding proactive forward ant to the next hop by applying the equation 6.14. When the node receives the ant, it consults its routing table if the field *disjoint session* has the same value as the source of the ant. In the case of having the same value, we discard the packet, because it is treated as a node disjoint route.

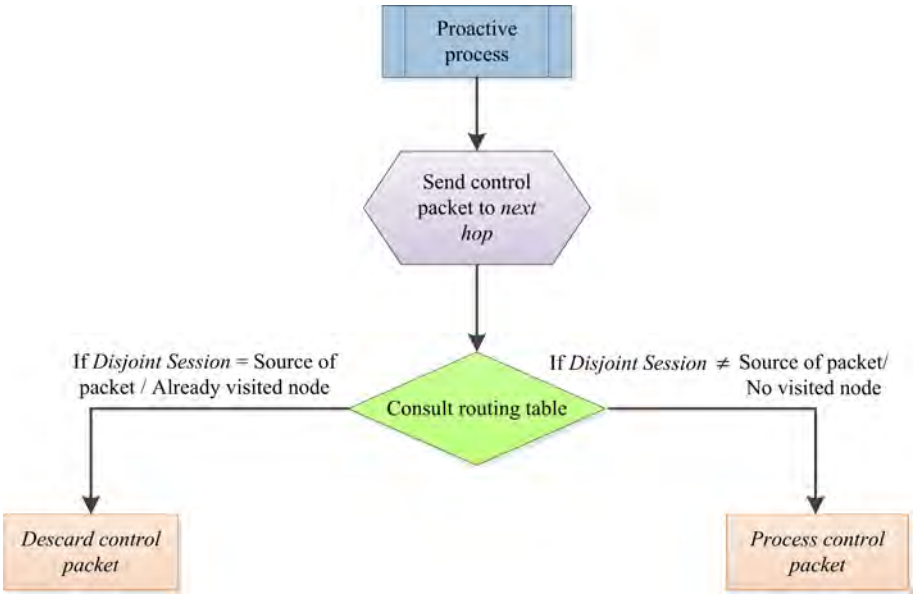


Figure 6.26: Flowchart of disjoint node routes (AntOR-DNR)

Algorithm 6.2 represents the pseudocode of calculation process of disjoint node routes.

```
while Proactive Process do
    {nexthop} = GetNextHop(dst);
    Send(nexthop,msg);
    {src,dst} = GetSessionInfo(msg);
    if CheckDisjointNode(src) = TRUE then
        Discard(msg);
    end
end
```

Algorithm 6.2: Calculation of disjoint node routes (AntOR-DNR)

Line 1 shows a proactive process that is represented with a loop to indicate that is done continuously after the initiation of the data session. Lines 2 and 3 obtain the next hop (*nexthop*) and proactive forward ant (control packet *msg*) it is sent. Line 4 gets the source *src* and destination *dst* of data session from this control packet. Line 5 checks the property disjoint node, i.e. it is verified if that source *src* is the same as the field *Disjoint Session* from the routing table. If affirmative case, line 6 is performed, discarding such a packet because it is a disjoint node route.

6.6 AntOR - Restrictive Disjoint Link Route (AntOR-RDLR)

As its name suggests, AntOR-RDLR is derived from the protocol AntOR-DLR, presenting two important differences with respect to this. The first, which gives rise to its name, is what occurs in the route established maintenance phase and exploration of new routes, where, on the one hand flexibility this by enabling proactive ants the coexistence of non-disjoint link routes with disjoint link routes, and on the other, it restricts these to contain a maximum number of disjoint links. The second difference occurs in the route setup phase and is related to the pheromone update process.

Then we look deeper into each of these differences.

The pheromone update process in the route setup phase is as follows: If the node i that has a route to the destination d already has a value of virtual pheromone and in the route setup phase we obtain other regular pheromone applying the equation 6.7, then the value of regular pheromone replaces the pheromone virtual using the maximum of these two values, and setting the value of virtual pheromone to 0. Equation 6.17 summarizes the process:

$$\begin{aligned} Regular_{last} &= F(Regular_{new}, time) \\ Regular_{final} &= max(Regular_{last}, Virtual_{old}) \\ Virtual_{final} &= 0 \end{aligned} \tag{6.17}$$

As discussed above, during the maintenance of established routes and exploration of new routes phases of AntOR-DLR the proactive forward ants do not use disjoint link routes. On the contrary, in AntOR-RDLR it is possible to choose disjoint links for data retransmission up to a maximum of MAX_HOP hops.

Algorithm 6.3 represents the route calculation process in AntOR-RDLR.

Line 1 indicates that a proactive process is continuously occurring. Line 2 gets the source *src* and destination *dst* of the data session using method *GetSessionInfo*. Line 3 constitutes the core of this algorithm. This line gets current allowed counter of hops *hop*, accessing a field of the message *msg* which contains information about the number of disjoint node that have been traveled by proactive ants. Lines 4 and 5 calculate the possible next hop *nexthop* to route the message *msg* as well as the possible link *link*. Line 6 checks if the link is disjoint or not. If it is not disjoint (line 7) corresponding proactive ant is sent to the next hop *nexthop*. In affirmative case, that is, if it is treated as a disjoint link, it applies property so-called restrictive, which consists of transmitting for disjoint link (non-allowable in AntOR-DLR) up to a maximum number of times MAX_HOP (line 8). Lines 9 and 10 update the value of current hop counter *hop*. Finally, it is allowed to send (line 11) the ant proactive using *nexthop* previously calculated.

```
while Proactive Process do
    {src,dst} = GetSessionInfo(msg);
    {hop} = GetAllowedHop(msg);
    {nexthop} = GetNextHop(dst);
    {link} = GetLink(dst,nexthop);
    if CheckDisjointLink(link) = FALSE then
        Send(nexthop,msg);
    else
        if hop ≤ MAX_HOP then
            hop = hop + 1;
            UpdateAllowedHop(hop);
            Send(nexthop,msg);
        end
    end
end
```

Algorithm 6.3: Route calculation (AntOR-RDLR)

In AntOR-DLR a selected link from a link disjoint route is not a candidate for sending in the retransmission process of proactive agents. AntOR-RDLR can retransmit by such a link up to a maximum number of attempts MAX_HOP. This is possible thanks to a field Reserved of the packet header. AntOR-DLR this field has different function: number of hops allowed in broadcast mode in the route local repair process. In AntOR-RDLR is utilized to indicate the current number of selection of a disjoint route in the exploration process of new paths.

Figure 6.27 shows an example of restrictive property.

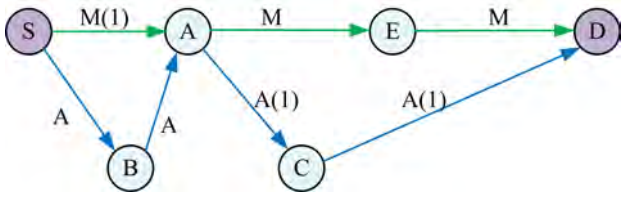


Figure 6.27: Example of proactive process (AntOR-RDLR)

The main route created is shown in green (label M) and blue color the possible alternative route (label A), that the proactive forward ants (PFA) can explore. We show the current counter of attempts cont as a number in parentheses after the tag. In this example the node S starts a proactive process of exploration of new route, sending the corresponding PFA to destination D. In AntOR-DLR proactive ant cannot go by links that belong to the main route (when we deal with disjoint link route). AntOR-RDLR allows a maximum number (MAX_HOP) of opportunities of choosing a link belonging to the route main. MAX_HOP is set to a value of 2 hops allowed. Finally, node S selects node A to relay the PFA. This is allowed because the attempt number of MAX_HOP has been set to 2 and the current counter cont has only utilized 1, which has not surpassed the set amount. The next node that is chosen is C. As the link formed by nodes A-C does not belong to the main route, the current counter cont is not increased. The same occurs for the link formed by the nodes C-D. When reaching this ant to the destination, as AntOR-DLR, the corresponding PBA is sent to update the entries from routing table of the route that has been indicated in the forward phase.

6.7 AntOR - Unicast Disjoint Link Route (AntOR-UDLR)

As its name suggests, AntOR-UDLR is derived from protocol AntOR-DLR, differing from this in the link failure management phase.

Previously to AntOR-UDLR specification should outline the differences in the unicast messages versus broadcast messages. Unicast means the sending of information from a single sender to a single receiver. Broadcast means the sending from a single sender to the network indiscriminately. Unicast mode has the advantage that it produces fewer collisions (and, consequently, less loss of messages), but it has an additional delay since it checks by means of control message that the channel is free to transmit.

AntOR-UDLR replaces notification messages sent in broadcast mode in AntOR-DLR by simple messages (unicast) sent to the predecessor which has a valid route to a reachable destination, understanding for valid route the belonging to the active session of a given destination with a positive value of regular pheromone. When a node detects a link failure in its neighbor, it communicates to its predecessor such a failure by means of a unicast message, repeating this process until it reaches the source node of the data session. This triggers the source to launch a forward ant for route setup. It may happen that the node that detects the failure has two or more overlapping data sessions, communicating this to source nodes of the involved different data sessions.

Figure 6.28 illustrates the notification process of failure of link in AntOR-UDLR:

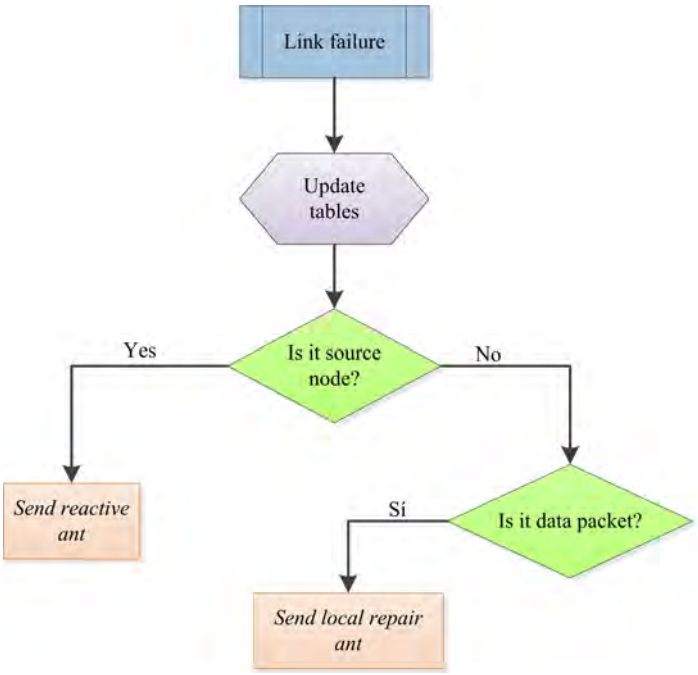


Figure 6.28: Link Failure Managemente (AntOR-UDLR)

When there is a node failure, both link and node failure occurs. The node that perceives the failure eliminates from its neighbor table to the corresponding node. Then, it updates the routing table with the new information of pheromone and proceeds as follows:

a) If there is no route at source, a reactive forward ant is sent.

b) If there is no route at intermediate node and it was a data packet that was retransmitting when the failure occurred, it sends a route repair forward ant. If there is no

reply from the corresponding repair backward ant in a determined period of time, a unicast message is sent to the precursor of the route informing that the destination is unreachable. The node that receives this message updates the routing table, and forwards this message to the predecessor and so on up to the source node of the data session.

c) If there is no route at the intermediate node, and if it is dealt with control packet (a *Hello* message or a unicast control message), any message is not sent. This can cause we have not repaired routes properly. When an intermediate node, which routes the data, does not find a valid route, it sends a unicast message to all one-hop neighbors to update their routing tables. It is necessary to send this message to all neighbors because, otherwise, upon not finding a valid route, there is no information of the predecessor. When one of these neighboring nodes has a valid route to the destination, it forwards the unicast message to the precursor of the route, and so on until reaching the source node.

Figure 6.29 shows the structure of *Unicast Link Notification* (ULN) message:

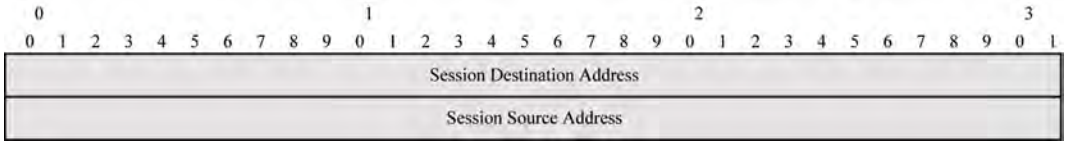


Figure 6.29: Unicast Link Notification format message (AntOR-UDLR)

As we can see, it has a simple structure, containing only two IP addresses: *Session Destination Address* and *Session Source Address*. The first address refers to the destination from the data session with a valid route and the second refers to the source. The destination address is used because when a link failure occurs, the node that detects it must indicate the destination so that predecessor nodes can properly process the message and decide whether they forward it in the case that there is a valid route to the destination. The source address is also required because it indicates to the node that receives the message if the source has been reached by checking if the source address that is encapsulated in the message is the same as the main address of the node.

Algorithm 6.4 shows the pseudocode of the neutralization process of link failure.

```
{src,dst} = GetSessionInfo(msg);
if CheckValidRoute(dst) then
    if CURRENT_NODE = src then
        SendRFA();
    else
        TTL = TTL - 1;
        {pre} = GetPrecursor(dst);
        Resend(pre,msg);
    end
end
```

Algorithm 6.4: Neutralization of link failure (AntOR-UDLR)

Line 1 obtains source and destination of the data session. This information is extracted from the message ULN. Line 2 checks if there is a valid route to the destination *dst* (session

active and positive value of regular pheromone). In affirmative case, it is checked (line 3) if the current node (the one which receives the message ULN) is equivalent to *src*. If the source of the data session has been reached a new route setup (line 4) is processed. In contrary case (line 5) message ULN (lines 6 to 8) is forwarded. Line 6 decreases the value of TTL field by one unit. This field is included in the packet header. Line 7 gets the predecessor *pre*, so that line 8 can do the forwarding of the ULN message.

Figure 6.30 illustrates an example that explains the way in which a link failure is treated at an intermediate node when a data message is transmitted and it is unable to repair the route (case b). The example network is formed by 5 nodes, being the source node and destination node A and E, respectively. The node that fails is marked in red, causing link failure between C and D. C notifies its predecessor B with a simple unicast message (ULN) that the destination E is unreachable. Upon receiving B this message B, it forwards it to predecessor A, decreasing the TTL value of such message in a unit. Finally, when the node source A receives this message a new route setup process is executed.

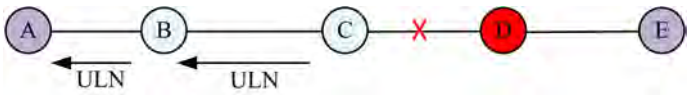


Figure 6.30: Example of link failure management - case b (AntOR-UDLR)

Figure 6.31 illustrates the case c previously mentioned that is specific of AntOR-UDLR. This Figure is formed by 6 nodes, being A and E the source and destination of a data session, respectively. According to Figure 6.31(a) Node A forwards the packet data (*Route*) to the reachable destination E through the next hop B. Upon receiving the data packet correctly, node B forwards it to node C with destination E. As now C does not find the route (*No Route*) to the next hop D, it cannot relay it, so the information cannot be routed to the destination successfully. At this moment the specific process of AntOR-ULDR is applied (see Figure 6.31(b)) by sending a unicast (ULN) message to the neighbors. To be able the corresponding message to send to the neighbors,it is necessary to search the IP addresses of each one of them in the neighbor table, by sending a unicast message for each IP address of the found neighbor. F and B nodes receive the message sent by C, but D not because it is eliminated from neighbor table of C, since it was what the failure originated. Node B forwards it to A, since it belongs to the data session (A, B, C, D, E). Upon receiving A this message, it sends a reactive forward ant to proceed with a new route setup. On the other hand, the node F processes the message, but it does not forward it because it did not belong to the valid route to the destination E.

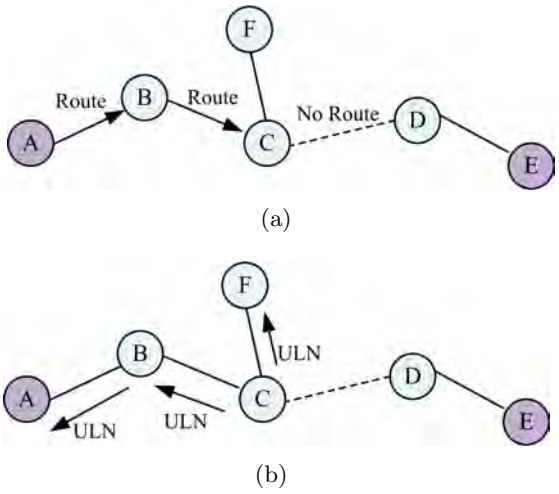


Figure 6.31: Example of link failure management - case c (AntOR-UDLR)

6.8 AntOR-v2

As its name implies, AntOR-v2 is derived from the protocol AntOR (more specifically of AntOR-DLR), although there are important differences, namely: control packet buffering, obsolete route management of, management of sending failures and removal of virtual pheromone in the maintenance of established routes and exploration of new routes phase. Then, these differences are discussed in detail.

The control packet buffering consists of these are stored for subsequent sending to their respective destinations at every certain interval of time. This fact allows it to have synchronism in the sending of packets and not to congest the network, decreasing its collision. Each entry in the buffer includes the following information: a) Socket which sends the packet; b) the control packet or particular message of the protocol; and, c) destination address (it can be a broadcast address or unicast address sent to a specific node).

The obsolete route management replaces the evaporation process of pheromone. This event takes place every certain interval of time and is as follows:

- Each entry in the routing table has a field (*timestamp*) indicating when it was created or was last updated such an entry.
- If the field *timestamp* associated with each route in the routing table is lesser than the difference between the current time and a given time limit, it is removed the aforementioned entry in a local way (each node).
- The value of this limit is important. A low value makes the routes converge slowly, eliminating routes to active destinations. Conversely, a high value implies a high convergence in the creation of routes with the consequent possibility of maintaining obsolete routes.

The management of sending failures is related to a fault tolerance. When a failure is detected, a neutralization process is launched. In highly dynamic environments (with more links breaks) the number of neutralization processes as a route local repair is greater, causing a major overload. The introduced mechanism pretends to alleviate this fact by

checking for the existence of valid path (positive value of regular pheromone) to the neighbor that will be transmitted. Only in the case that the path exists, the control packet is sent.

The fourth and final difference and, perhaps the most significant is, as mentioned above, the elimination of virtual pheromone in the maintenance of established routes and exploration of new routes phase. It is intended to reduce the overhead using proactive agents that do not require routes with virtual pheromone. These agents create alternative routes that go from neighbor to neighbor until reaching the destination node. At the time of selecting the next hop, the agents take into account the maximum value of regular pheromone from one-hop neighbor. In this way we can reach alternative routes, which are also link disjoint. These proactive ants are sent when the number of alternative routes is less than a certain threshold.

Figure 6.32 shows a selection example of the next hop in the exploratory proactive process. The main route (A, B, E) of red color is created in the route setup phase. In the exploration phase, node A sends corresponding PFA, having to choose between intermediate 3 neighbors: B, C, D. These neighbors have values of regular pheromone of 20, 5 and 15, respectively. These values of pheromone are inversely proportional to the time estimate generated by the reception of *Hello* messages. B is the best candidate to forward (higher value of pheromone), but it belongs to the main route, so then the next best are chosen (in this case the intermediate node D). This process continues through the intermediate nodes to reach the destination node. Finally, it is worth noting that the variable MAX_TTL (maximum TTL) from the PFAs controls the maximum number of hops in the alternative routes.

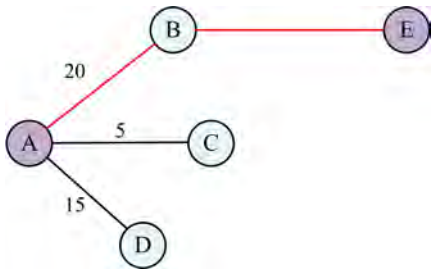


Figure 6.32: Example of proactive process (AntOR-v2)

6.9 Hybrid ACO Routing (HACOR)

HACOR consists of a refinement of AntOR-v2, differing from this in the incorporation of the data packet buffering capacity, in a optimized link failure neutralization process and the introduction of a particular type of S-ACO during maintenance of established routes and exploration of new routes. Then, these differences are discussed in detail.

The data packet buffering consists of storing these for subsequent sending to their respective destinations at every certain interval of time on the assumption that there are no routes. Indeed, when the data packet is ready to send to the next hop, it checks if there is a valid route to the destination belonging to the current data session. In the case that there is not a valid route, the data packet is stored in packet queue, sending a local route repair forward ant to solve the problem. At the same time, it tries to repair the route, the node sends a unicast message to all reachable neighbors. The neighbors, which receive this message, send it to its predecessors. Otherwise, that is, if there is a valid route, it is

processed to the sending.

Algorithm 6.5 shows the pseudocode of the neutralization process of link failures.

```
if CheckLink() = TRUE and TransmissionError() = TRUE then
    UpdateNeighbor() ;
    DeleteAllRoutes() ;
    if CheckSource() = TRUE then SendRFA();
    else if CheckData() = TRUE then SendLocalRepairAnt();
    else if CheckHello() = FALSE then SendUnicastPrecursor( );
end
```

Algorithm 6.5: Link failure management (HACOR)

The first event that occurs when there is a node failure is that the node which perceives it, updates its neighbor table, eliminating all routes that have the node that fails as a next hop.

If there is not a route at the source node, it starts the route setup sending a reactive forward ant.

If there is no route at an intermediate node and a data packet was being forwarded when the failure occurred, it is sent a local route repair forward ant to each of the destinations of all affected data sessions.

If there is not route at the intermediate node and it was being sent a control packet (Hello) in broadcast mode, any neutralization process is not performed.

If it was being forwarded a unicast control packet, a ULN message is sent to the predecessor node. This process is repeated until we reach the source node.

Finally, the third distinctive feature of HACOR with respect to its predecessor is the introduction of a variant of S-ACO in the maintenance of established routes and exploration of new routes phase which basically consists of the following:

a) The virtual pheromone leaves to be necessary at this phase.

b) We do not utilize the evaporation process.

c) A free-loop method is used (see Figure 6.33) when proactive forward ant (PFA) has come to the destination node. Subsequently the loop is removed, making this PFA in a free-loop PBA, which returns to the source by the visited nodes in the list, updating the routing tables of each node.

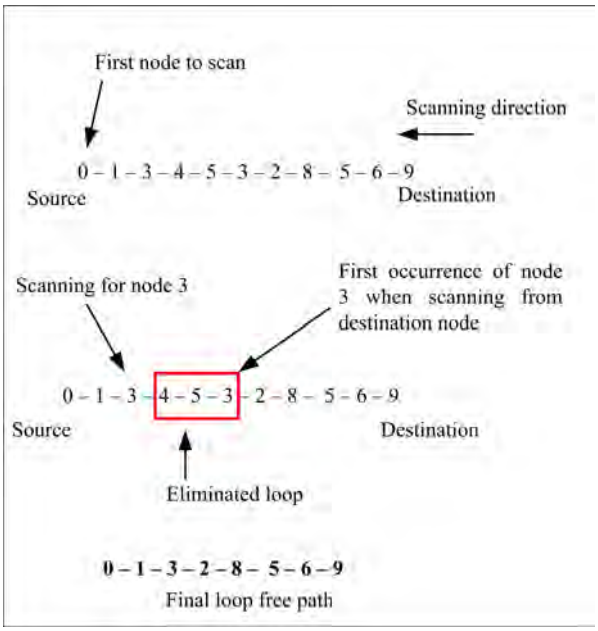


Figure 6.33: Loop elimination process (HACOR)

d) We do not need the initial establishment of pheromone values to each one-hop neighbor. The exploration process is done hop-by-hop with the pheromone information that have the one-hop neighbors using the Hello messages. Each node that receives a Hello message from another one-hop neighbor updates its route with the new value of pheromone.

e) The proactive forward ants utilize link-disjoint route. Figure 6.34 presents a scheme of this exploratory process from the source node to the destination node.

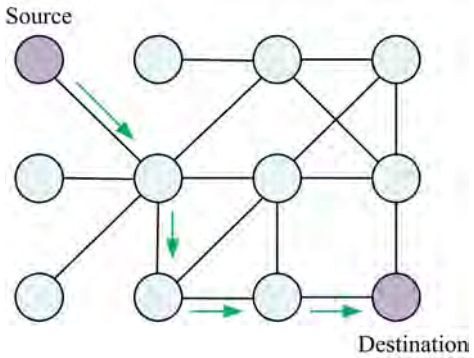


Figure 6.34: Example of path exploration (HACOR)

f) This use of disjoint routes involves the checking of whether the one-hop neighbor that has to forward the corresponding proactive agent belongs to a disjoint route or not. In the case that the neighbor belongs to a route disjoint, it is not chosen (in order to reduce the overhead).

6.10 Parallel AntOR (PAntOR)

As its name suggests, PAntOR derives from AntOR, more specifically, it can be considered a parallel approximation of AntOR-DNR. The reason for choosing AntOR-DNR (and not AntOR-DLR) is that it is intended to analyze the worst case, hence the choice of the first for being more restrictive.

Previously to PAntOR specification it is advisable to point out some aspects of the parallelization of the ACO algorithms.

Firstly, it should be know that practically all parallel work on ACO algorithms are designed for centralized systems based on technical master - slave, where the central master distributes work to other processors. PAntOR, on the other hand, is designed for systems decentralized, which gives it even greater relevance.

Secondly, good to know that the parallel ACO algorithms are classified according to the two criteria are described below:

A possible classification differences if a parallelization of an ACO algorithm is standard or is especially designed. A standard ACO parallelization aims to decrease run time without changing the behavior of the algorithm. On the contrary, specific parallel algorithms change ACO in order to obtain a more efficient algorithm. A method utilized to differentiate between these two approaches consists of making use of the exchange of information between processors.

Another possible classification checks if the algorithm has a centralized or decentralized approach. In a centralized approach, it is normal that the processor collects information of pheromone, as well as the different solutions of other processors. Thus the pheromone update is done in a central manner. In a decentralized approach each processor has to calculate the pheromone update itself using the information received from other processors.

PAntOR consists of a standard ACO parallelization (large-grained parallelization) with a decentralized approach.

To understand how PAntOR works, it is necessary to employ three concepts:

a) *Process*: Program running. The processes are managed by the Operating System.

b) *Thread*: The basic unit of execution. Any program that executes at least has a thread.

c) *Portable Operating System Interface (POSIX) Thread*: Standard based in threads *Application Programming Interface* (API) for C/C++.

We use POSIX Thread because it allows a new concurrent process flow to expand. This is the most efficient multi-core systems, where the flow of processes can be scheduled to run on another processor, thus gaining speed through parallel or distributed processing. Programming with threads carries less overhead than expanding a new process, because the system does not initialize a new environment and virtual memory space for that process.

Parallel programming technologies, such as MPI and *Parallel Virtual Machine* (PVM), are used in a distributed computing environment, while the threads are limited to a single computer system. All threads within a process share the same address space. For the implementation of this routing algorithm to be faster, we use the POSIX Thread library.

This parallel technique involves launching a thread for each neighbor that initiates any of the following processes: route setup, local route repair and link failure notification.

Figure 6.35 shows the flow chart of parallelism introduced in the route setup process. This process will parallelize by means of threads, sending a reactive forward ant to

neighbors which are one-hop using a separate thread, being the number of threads utilized proportional to the number of neighbors from node that initiates this process. When an intermediate node receives this ant repeat the process. On the other hand, if it is a destination node, this sends its corresponding reactive backward ant (RBA).

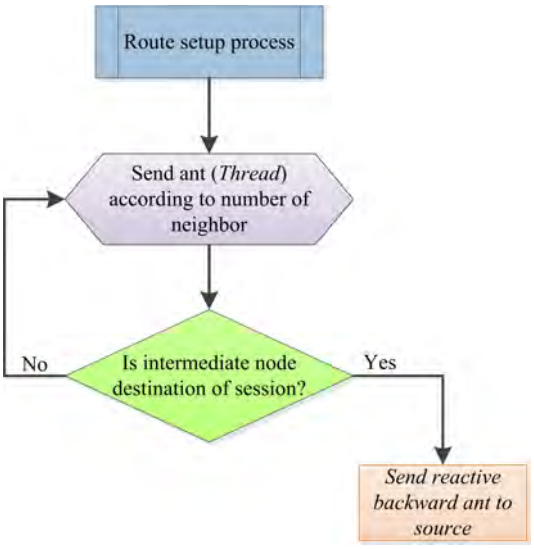


Figure 6.35: Parallelization of route setup process (PAntOR)

Figure 6.36 shows the flow chart of parallelism introduced in the local route repair process. As you can see, its functioning is analogous to the mentioned in the route setup process, unless it is done in the local level.

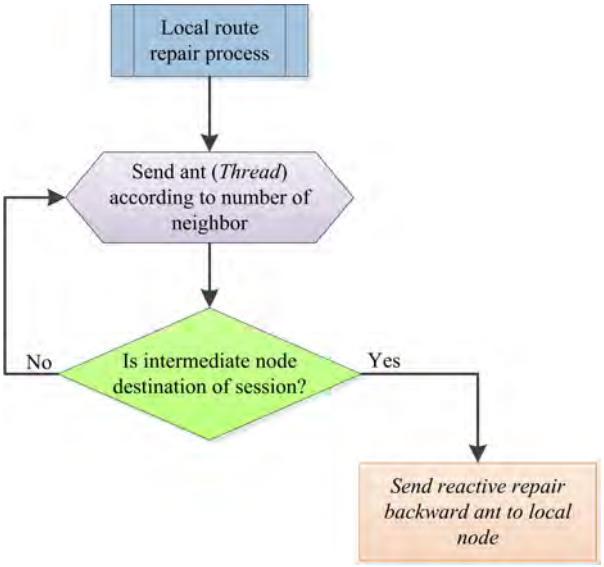


Figure 6.36: Parallelization of local route repair process (PAntOR)

Figure 6.37 shows the flow chart of parallelism introduced in the process of link failure notification. As already mentioned in subsection 6.3.4, this process aims to update the routing table in face of link failures. It is a phase of great importance, and it is crucial

that is it performed quickly. The ants send in independent threads until an intermediate node has some alternative route to the destination after updating the routing table.

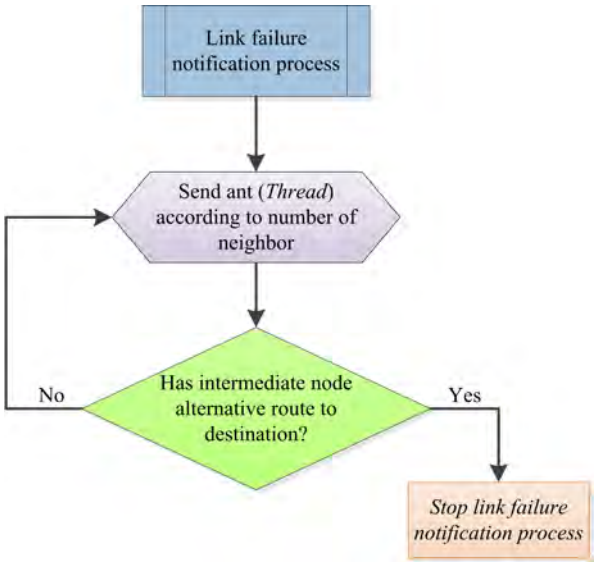


Figure 6.37: Parallelization of link failure notification process (P-AntOR)

Figure 6.38 shows an example of functioning of PAntOR. If node A wants to start the route setup process in AntOR, it consults the candidates to send a reactive forward ant in its neighbor table N = $\{N_1, N_2, N_3\}$in an independent thread. It is sent 3 threads in PAntOR.

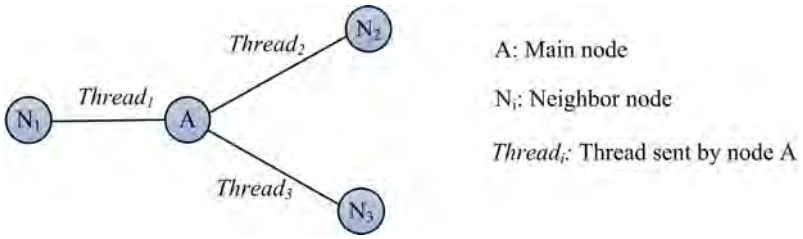


Figure 6.38: Example of functioning (PAntOR)

6.11 PAntOR - Multiple Interface (PAntOR-MI)

As its name suggests, PAntOR-MI is a variant of PAntOR designed for devices that contain more than one interface, i.e., for small and portable devices with more than one antenna or wireless network interface (PocketPC, mobile phones of last generation and so on).

PAntOR-MI parallelizes the sending ants broadcast through the interfaces using threads. Due to the difficulty of finding specialized hardware, PAntOR-MI only has been applied to the route setup process using two interfaces.

Algorithm 6.6 shows the route setup process in PAntOR-MI. As you can see, while running the route setup process, a reactive message is sent in broadcast mode by each interface having the node, managing such an interface through a thread.

```
while Route Setup Process do
    for Cont = 1 to Max_Interfaces do
        Send Broadcast Message by Thread(Cont);
    end
end
```

Algorithm 6.6: Route setup (PAntOR-MI)

Figure 6.39 shows a functioning example of PAntOR-MI. We assume that the medium (wireless channel) is the same for all devices, and all nodes are homogeneous (equal computational capacities and with identical ranges of transmission).

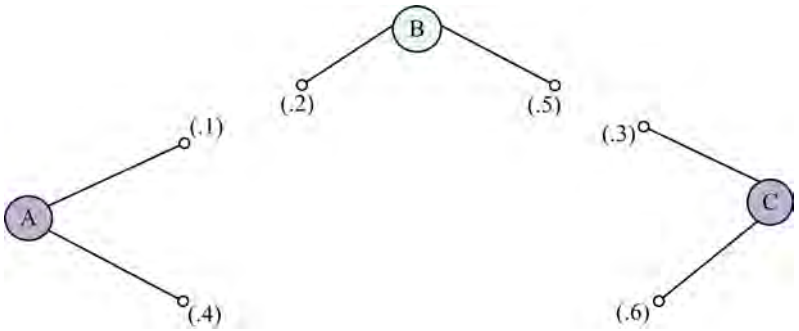


Figure 6.39: Functioning PAntOR-MI

In this example we have 3 nodes (A, B, and C), which have with two network interfaces. The source node is A and the destination node is C. Each network interface has associated a unique IP address. Each node considered main address to one of these IP addresses. The main addresses to nodes A, B and C are (. 1), (. 2) and (. 3) respectively. Each network interface has associated its corresponding main address and each node stores the following information for its two interfaces: (*IP Address of the Interface,Main IP Address*). For example, node A stores the following information for interface: (.1, .1) and (.4, .1).

Data packets are retransmitted by the main addresses and the functionality of PAntOR-MI consists of managing through a thread of sending RFAs. They are sent in broadcast mode not to divide the created routes in the route setup process and to ensure that the RFA message sent by an antenna (interface output) of node arrives (if it is not the case, it check that is received by other interface). If every interface of the same node receives an RFA, it discards it because it is treated of the same node. On the other hand, in the intermediate nodes it is not discarded because we check that its main address does not correspond to its interface.

The RFA in its list of visited nodes are storing each main IP address associated to each visited interface. Upon arriving the RFA the destination node C, it is processed the information from the RFA, becoming a RBA that returns hop by hop in unicast mode to the source node A, utilizing the information learned from the addresses stored in the list of visited nodes. In this return process the routes in the intermediate nodes will be updated or created with the main addresses. This approach takes into account only the main addresses because the data are routed by them and it aims to create routes as soon as possible in the setup process.

6.12 Summary

The main objective of this chapter has been the specification of a family of ACO routing protocols for mobile ad hoc networks. These protocols have a common root: the protocol AntOR, inspired in AntHocNet, which inherits its hybrid character, multipath and adaptive. We have started to see the use of disjoint node and / or link routes, the separation between the virtual and the regular pheromone and some changes in the maintenance of established routes and exploration of new routes phase are the main differences of AntOR with respect to its predecessor. Subsequently, we have presented data structures from protocol and we have described the four phases of the same in detail. Then, we have shown specific protocols that derive from protocol base AntOR, seeing in first place the variants AntOR disjoint link (AntOR-DLR) and AntOR disjoint node (AntOR-DNR) that utilize routes that do not share links/nodes, respectively. Then we have shown how these variants give rise to other protocols by successive refinements. More specifically, we have seen how AntOR-RDLR, AntOR-UDLR, AntOR-v2 and HACOR derived from AntOR-DLR. Thus, AntOR-RDLR differs from AntOR-DLR in the pheromone update process and the mechanism of route exploration; AntOR-UDLR is an approximation unicast of AntOR-DLR replacing its link failure notification messages, which are sent in broadcast mode, by unicast messages to the predecessor of the node that reports the link failure; AntOR-v2 incorporates control packet buffering and management of obsolete routes and failures of sending, eliminating the use of virtual pheromone in the new route exploration phase; and HACOR also incorporates the data packet buffering, optimal failure neutralization and the use of S-ACO in the route exploration. Finally, we also have seen how PAntOR and PAntOR-MI derived from AntOR-DNR: PAntOR is a standard ACO parallelization of protocol AntOR-DNR with a decentralized approach and PantOR-MI is a version of PAntOR designed for multi-interface devices.

Chapter 7

Simulations and Results

This chapter presents the simulations carried out utilizing various real scenarios that prove the applicability of the different proposals. To achieve this, we have used the network simulator NS-3 [NS3], one of the most utilized in the field. First, we discuss the selection of the chosen network simulator. Then we describe the simulation scenarios used. Subsequently, we discuss the metrics that we have analyzed as, for example, throughput, delivered packet ratio, average End-to-End delay, jitter, overload in the number of packets, overload in number of bytes and so on. The chapter ends with a brief summary of what has been shown.

7.1 Election of Network Simulator

The simulations are utilized as support in the protocol design. There are two important aspects that should be evaluated prior to carrying it out: use the appropriate model and make the right choice of the best tool for the model in question.

Then we present the more relevant network simulators as well as the characteristics for each of them:

- ***Network Simulator 2*** **(NS-2)**: *Network Simulator 2* (NS-2) [NS2]is a discrete-event network simulator targeted primarily for research and educational purposes. The simulations are composed of code written in C++ (which is used to model the behavior of simulated nodes) and command scripts *Object-Oriented Tool Language* (oTcl) (which control the simulation and specify additional aspects such as the topology of the network). This design was chosen to avoid unnecessary recompilations when changes were made to the structure of the simulation since a frequent recompilation of the C++ program consumed much time when the first version came. However, this is not currently a problem and it is not necessary to sacrifice the performance of simulation to save on recompilations, especially when a great network is simulated [BHvR05].

- ***Network Simulator 3*** **(NS-3)**: Similar to its predecessor, NS-3 [NS3] is a discrete-event simulator and is based on C++ for the implementation of models of the simulation. However, NS-3 does not use command scripts oTcl anymore to control the simulation avoiding the problems presented by the combination of C++ and oTcl in NS-2. Simulation scenarios in NS-3 can be implemented in pure C++ and, optionally, parts of the simulation can be done using Python.

- ***Objective Modular Network Testbed in C++* (OMNeT++)**:In contrast to NS-2 and NS-3, *Objective Modular Network Testbed* (OMNeT) [OMN] is not a network simulator by definition, but a general-purpose discrete-event based simulator. However, it is applied above all to the domain of network simulation, taking into account the fact that its package *Integrated Network Enhanced Telemetry* (INET) offers a wide model collection of Internet protocols. The simulations consist of simple modules that behave like a model, for example, a specific protocol. Several simple modules can be connected to form a composite module [BHvR05]. Like NS-2 and NS-3, OMNeT++ is based on C++ for the implementation of simple modules. The composition of these simple modules in composite modules and, therefore, the simulation setting, takes place in *Network Description* (NED), network description language of OMNeT++.

The network simulator mentioned, the most utilized is NS-2 in the academia and research area. However, many users complain about the complexity of the simulator and the high consumption of resources leading to lack of scalability, impeding the execution of network simulations with hundreds of nodes [Kök08]. This is because the simulation time increases exponentially with the number of nodes in the network and also consumes lots of memory to execute the simulation.

Due to all these problems, NS-3 is created. One of its main objectives was to eliminate the problem of scalability and support the simulation of parallel and distributed [HRFR06]. In spite of the fact that NS-3 does not have all the models that currently NS-2 has, it has more details of the IEEE 802.11 standard models and it is possible to integrate new modules making it possible to update the simulator, allowing it to continue to the rapid growth of wireless networks [HRFR06]. In addition, NS-3 has new features such as: correct handling of multiple interfaces, usage of IP addresses, it generates PCAP files that are utilized for the analysis and so on.

With regard to OMNeT++ is a well organized, flexible and easy to use simulator. However, it has enough poor reports in the simulation results, so the users must develop code to obtain the desired metrics. It has external extensions which allow it to provide support for the simulation of wireless networks. However, it is only possible to simulate some scenarios since these extensions are not complete, especially that of mobility, in addition, the documentation is still under development and the analysis of performance metrics is deficient.

[WvLW09] demonstrates that OMNeT++ requires more time than NS-3 to perform a simulation, whereas NS-2 does not scale well and, therefore, is not suitable for simulations of great scale networks. Likewise, NS-3 is the most efficient simulator with respect to the use of the memory.

The above considerations determined the choice of the Network Simulator NS-3.

7.2 Simulation Environment

For the realization of the simulation of ACO routing protocols the following generic scenario have been considered:

- All nodes are configured at the physical layer using the standard IEEE 802.11b with a transmission range of 300 m.
- In the application layer *Constant Bit Rate* (CBR) is used to generate the traffic of each data session.

- The distribution of nodes is random.

- Utilized mobility pattern is *Random Waypoint* (RWP). In this model, the nodes move to destinations according to the randomness of this pattern RWP and, once the destination is reached, the nodes stop, according to the established time of pause, then continue to select another destination moving.

- The performed experiments are grouped into three types: variation in the scenario of the pause time, the number of nodes and speed.

Table 7.1: AntOR-DLR parameters

Parameter	**Value**
Number of nodes	[20 - 100] nodes.
Node distribution	Random.
Dimensions of area	1400 m × 1400 m.
Time simulation	30 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	3
Traffic generator	CBR.
Beginning of time CBR *client*	0 s.
Ending of time CBR *client*	30 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	30 s.
Number of data sessions	4
Data rate	2048 bits/s (4 packets of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[0 - 10] m/s.
Pause time	5 s.

Tables 7.1 to 7.16 present the parameters of the scenarios and protocols, respectively, utilized during the simulations.

Table 7.2: AntOR-DNR parameters

Parameter	**Value**
Number of nodes	100 nodes.
Node distribution	Random.
Dimensions of area	1000 m $\times$ 1000 m.
Time simulation	120 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	3
Traffic generator	CBR.
Beginning of time CBR *client*	0 s.
Ending of time CBR *client*	120 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	120 s.
Number of data sessions	5
Data rate	2048 bits/s (4 packets of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[0 - 10] m/s.
Pause time	[0 - 120] s with intervals of 30 s.

Table 7.3: AntOR-RDLR parameters

Parameter	**Value**
Number of nodes	100 nodes.
Node distribution	Random.
Dimensions of area	1000 m $\times$ 1000 m.
Time simulation	120 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	3
Traffic generator	CBR.
Beginning of time CBR *client*	0 s.
Ending of time CBR *client*	120 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	120 s.
Number of data sessions	5
Data rate	2048 bits/s (4 packets of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[0 - 10] m/s.
Pause time	[0 - 120] s with intervals of 30 s.

Table 7.4: AntOR-UDLR parameters

Parameter	**Value**
Number of nodes	100 nodes.
Node distribution	Random.
Dimensions of area	3000 m × 1000 m.
Time simulation	300 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	5
Traffic generator	CBR.
Beginning of time CBR *client*	Distribución Uniforme [0 - 60] s.
Ending of time CBR *client*	300 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	300 s.
Number of data sessions	10
Data rate	512 bit/s (1 packet of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[2 - 10] m/s with intervals of 2 m/s.
Pause time	[0 - 240] s with intervals of 60 s.

Table 7.5: AntOR-v2 parameters

Parameter	**Value**
Number of nodes	[50 - 150] nodes.
Node distribution	Random.
Time simulation	300 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	10
Traffic generator	CBR.
Beginning of time CBR *client*	Uniform distribution [0 - 60] s.
Ending of time CBR *client*	300 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	300 s.
Number of data sessions	10
Data rate	512 bit/s (1 packet of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[0 - 8] m/s.
Pause time	[0 - 240] s with intervals of 60 s.

Table 7.6: HACOR parameters

Parameter	**Value**
Number of nodes	[50 - 150] nodes.
Node distribution	Random.
Time simulation	900 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	10
Traffic generator	CBR.
Beginning of time CBR *client*	Uniform distribution [0 - 180] s.
Ending of time CBR *client*	900 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	900 s.
Number of data sessions	10
Mobility pattern	RWP.
Node speed	5 m/s
Pause time	[0 - 240] s with intervals of 60 s.

Table 7.7: PAntOR parameters

Parameter	**Value**
Number of nodes	100 nodes.
Node distribution	Random.
Dimensions of area	1200 m × 1200 m.
Time simulation	120 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	3
Traffic generator	CBR.
Beginning of time CBR *client*	0 s.
Ending of time CBR *client*	120 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	120 s.
Number of data sessions	5
Data rate	2048 bits/s (4 packets of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[0 - 10] m/s with intervals of 2.5 m/s.
Pause time	[0 - 120] s with intervals of 30 s.
Number of cores	4
RAM memory	4 GBytes
Parallel system	Threads through POSIX Thread standard

Table 7.8: PAntOR-MI parameters

Parameter	**Value**
Number of nodes	100 nodes.
Node distribution	Random.
Dimensions of area	1200 m × 1200 m.
Time simulation	120 s.
Physical layer	IEEE 802.11
Tranmission range	300 m
Number of runs	3
Traffic generator	CBR.
Beginning of time CBR *client*	0 s.
Ending of time CBR *client*	120 s.
Beginning of time CBR *server*	0 s.
Ending of time CBR *server*	120 s.
Number of data sessions	5
Data rate	2048 bits/s (4 packets of 64 Bytes per second).
Mobility pattern	RWP.
Node speed	[0 - 10] m/s with intervals of 2.5 m/s.
Pause time	2
Number of cores	4
RAM memory	4 GBytes
Parallel system	Threads through POSIX Thread standard

Table 7.9: Internal characteristics of AntOR-DLR

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Parameter 6 β_3	2
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	5
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2

Table 7.10: Internal characteristics of AntOR-DNR

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Parameter 6 β_3	2
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	5
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2

Table 7.11: Internal characteristics of AntOR-RDLR

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Parameter 6 β_3	2
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	5
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2

Table 7.12: Internal characteristics of AntOR-UDLR

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Parameter 6 β_3	2
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	3
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2

Table 7.13: Internal characteristics of AntOR-v2

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	3
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2
Limit time in obsolete route management	5

Table 7.14: Internal characteristics of HACOR

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	3
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2
Limit time in obsolete route management	5

Table 7.15: Internal characteristics of PAntOR

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Parameter 6 β_3	2
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	5
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2

Table 7.16: Internal characteristics of PAntOR-MI

Parameter	**Value**
Parameter 1 γ	0.7
Parameter 2 α	0.7
Parameter 3 η	0.7
Parameter 4 β_1	20
Parameter 5 β_2	20
Parameter 6 β_3	2
Maximum number of destinations in the HELLO message	10
HELLO emission interval	1 s.
PFA emission interval	2 s.
Number of retry for restoring the route	5
Number of broadcast allowed by RRFA message	2
Number of consecutive HELLOs may be loss	2

7.3 Performance Metrics

The performance metrics to evaluate the routing protocols are divided into metrics of effectiveness and efficiency. Effectiveness metrics are considered external measures to the protocol, because they measure if its performance is expected at the time of performing the task for which it was designed. As effectiveness measures are distinguished: throughtput, delivered data packet ratio, average End-to-End delay and jitter. On the other hand, efficiency metrics are considered internal and refer to the generated overhead. We highlight the overhead in the number of packets and the overhead in the number of bytes.

Defined performance metrics for the evaluation of designed ACO routing protocols were as follows:

- **Throughput**: Volume of work or information flowing through a system. It is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.
- **Delivered data packet ratio**: Relationship between the number of packets delivered successfully and number of packets sent.
- **Average end-to-end delay**: Average Time of the transmission of a data packet in the network from the source to destination.
- **Jitter**: Measurement of the variation of the time of arrival of consecutive data packets. This metric, classified as robustness and adaptability, is important in the QoS applications.
- **Overhead in the number of packets**: Relationship between the number of transmitted control packets and the number of delivered data packets successfully.
- **Overhead in the number of bytes**: Relationship between the total number of bytes sent, and the number of bytes of the delivered data packets correctly.

7.4 Evaluation of AntOR-DLR Protocol

To evaluate the performance of protocol AntOR-DLR, both in terms of efficiency and effectiveness, the impact of the increase in the number of nodes in the network has been taken into account (used the same area of simulation by varying the density of nodes), that is, how has an effect on parameters as the throughput, the delivered data packet ratio, average end-to-end delay, overhead in the number of packets and overhead in the number of bytes. This evaluation was developed jointly with the protocol AntHocNet.

7.4.1 Throughput

As shown in Figure 7.1, the throughput in AntOR grows in a slightly linear manner with the number of nodes. Likewise, we observe how in dense networks overcomes amply to AntHocNet, besides being much more stable than this. The reason for this improvement in the throughtput is due to increasing the number of nodes, it also increases the number of reliable alternative routes. All of the above can infer in the scalability of the protocol.

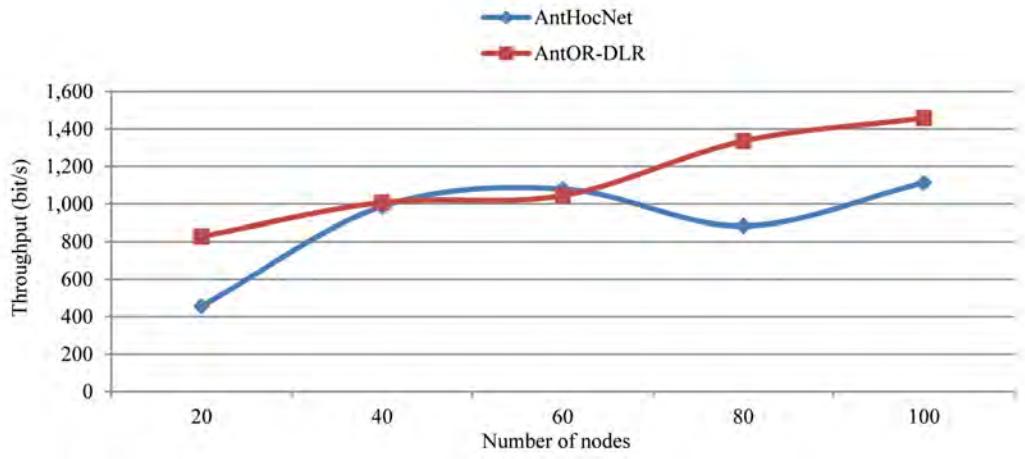


Figure 7.1: Throughput (AntOR-DLR)

7.4.2 Delivered Data Packet Ratio

As shown in Figure 7.2, the delivered data packet ratio in AntOR has a behavior analogous to the throughput.

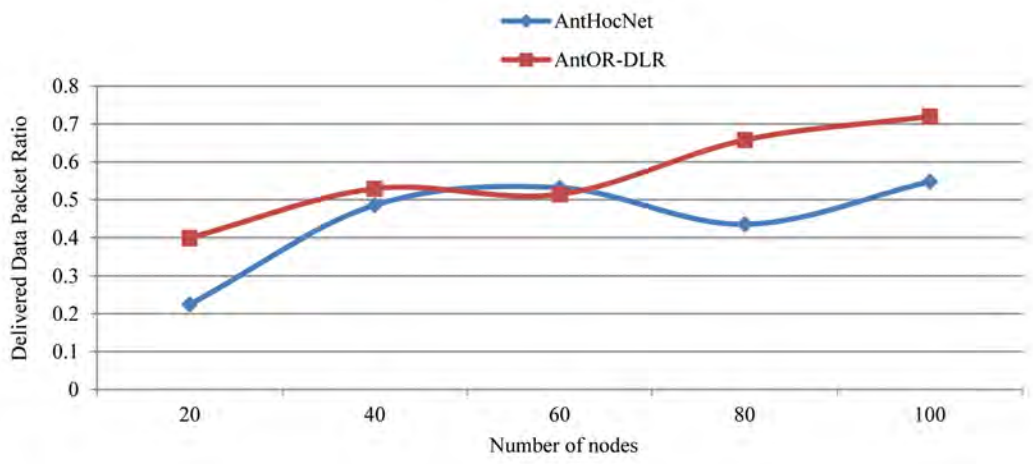


Figure 7.2: Delivered data packet ratio (AntOR-DLR)

7.4.3 Average End-to-End Delay

As shown in Figure 7.3, the average end-to-end delay in AntOR is greater than AntHocNet, being this difference almost imperceptible (the scale is in milliseconds) in dense networks. In addition, another noteworthy aspect is that, in general terms, the average End-to-End delay in AntOR is more stable than the corresponding to AntHocNet. This difference in the average end-to-end delay (which is reduced as the number of nodes increases) is a consequence of the fact that the link disjoint mechanism needs a minimum number of nodes to be effective.

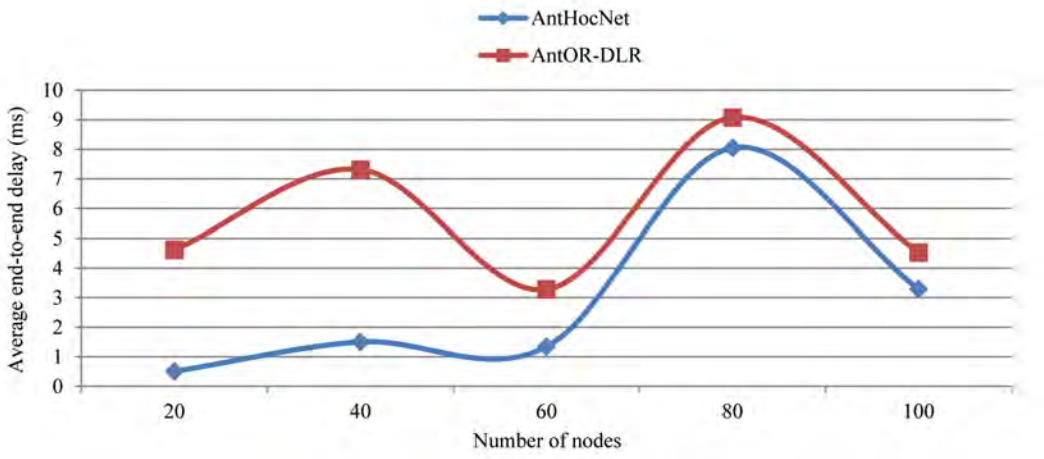


Figure 7.3: Average end-to-end delay (AntOR-DLR)

7.4.4 Overhead in Number of Packets

As shown in Figure 7.4, the overhead in the number of packets in AntOR is similar to the AntHocNet in little dense networks and lower in dense networks, so much when the number of existing nodes is greater. This decline in overload is explained by reaching the threshold number of nodes so that the link disjoint mechanism to be effective, the protocol tolerates much better failures (by having more reliable alternative routes) decreasing the number of control packets RRFA in the route repair process. This fact joined to the observed in the previous metrics allows us to conclude that AntOR is especially scalable, at least when compared with his predecessor AntHocNet.

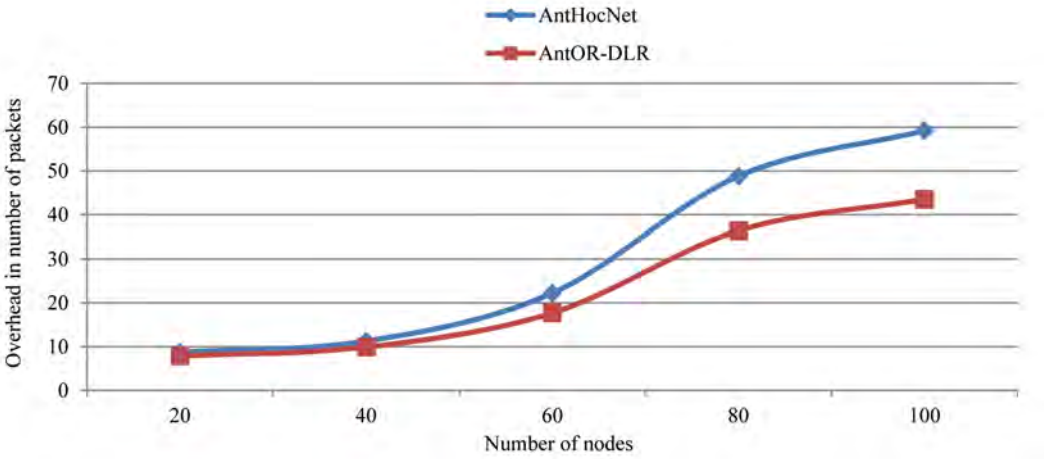


Figure 7.4: Overhead in number of packets (AntOR-DLR)

7.4.5 Overhead in Number of Bytes

As shown in Figure 7.5, the overhead in the number of bytes in AntOR behaves analogous to the overload in the number of packets, being able to conclude the indicated

previously.

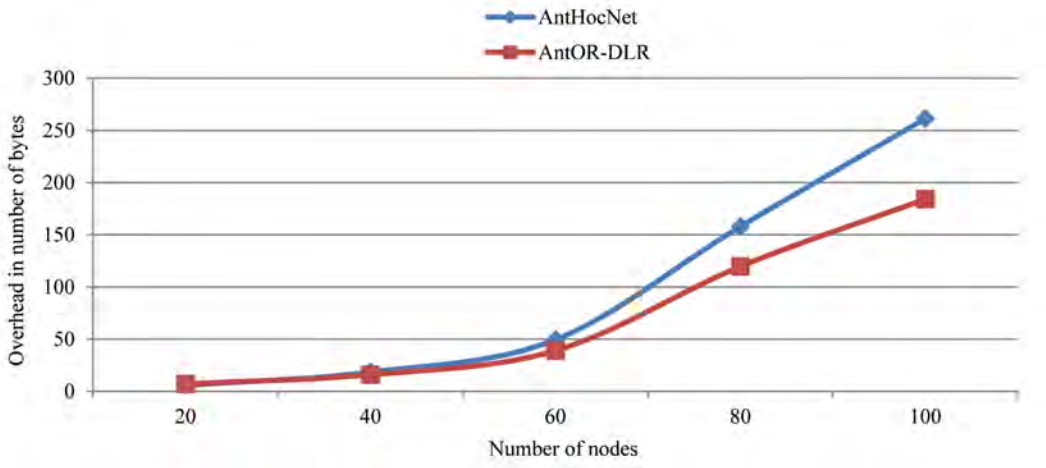


Figure 7.5: Overhead in number of bytes (AntOR-DLR)

7.5 Evaluation of AntOR-DNR Protocol

To evaluate the performance of the AntOR-DNR protocol, in terms of effectiveness, the impact of the increase of the pause time has been taken into account, namely, how this affects parameters as the delivered data packet ratio, average end-to-end delay and jitter. This evaluation was developed jointly with the AntOR-DLR protocol. It is worth mentioning that the variation of the pause time influences the behavior of the mobility pattern. This increase of the pause time has two effects in the general properties of the relevant scenarios for routing. The first effect is that the decrease in the mobility of the nodes (the result of a high pause time) makes the processing of the routing algorithm less difficult. The second effect which it has is related with the distribution of nodes in the area of the scenario when the RWP mobility model is utilized. It is proved that, according to this model, there is a tendency for the nodes to increase their density in the center of the network area and decrease at the extremes, especially when mobility is lower.

7.5.1 Delivered Data Packet Ratio

As shown in Figure 7.6, the delivered data packet ratio in AntOR-DLR is, at all times, greater than of AntOR-DNR, presenting also a monotonous behavior more stable. This is due to AntOR-DLR is less restrictive (more tolerant) than AntOR-DNR (see paragraph 6.3.3.3). In other words, the calculation of link disjoint route is easier (all disjoint node route is link but not vice versa) and also more frequent failure in disjoint node route (since the link disjoint route, that it is served of independent links, can use other nodes).

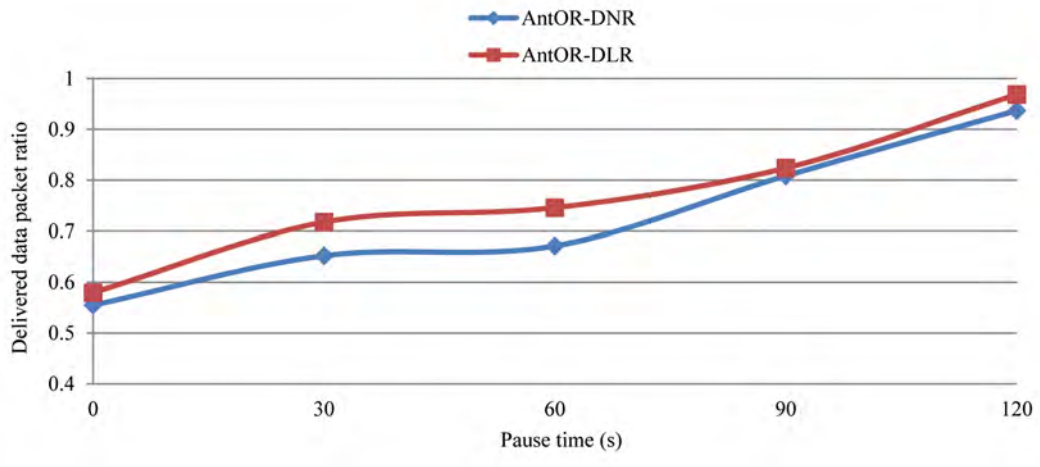


Figure 7.6: Delivered data packet ratio (AntOR-DNR)

7.5.2 Average End-to-End Delay

As shown in Figure 7.7, the average end-to-end delay is, at all times, less than for AntOR-DNR, presenting also a monotonous behavior more stable. The explanation of this fact is similar to the one carried out in the previous subsection.

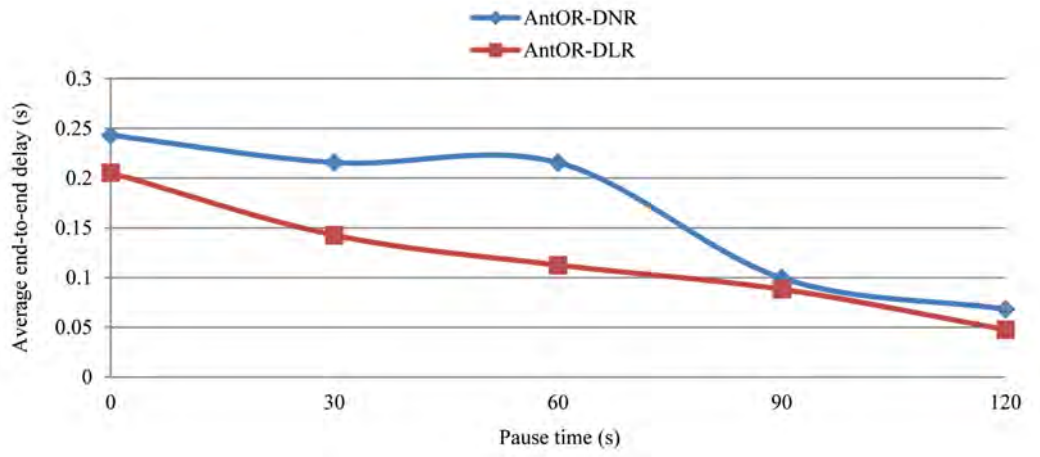


Figure 7.7: Average end-to-end delay (AntOR-DNR)

7.5.3 Jitter

As shown in Figure 7.8, the jitter in AntOR-DLR is, at all times, less than for AntOR-DNR. Unlike the two previous metrics the difference in the monotonous behavior of both protocols is more pronounced. It should be recalled that the jitter is a parameter that directly measure the robustness (behaviour versus fault) of the algorithm. Therefore, it is concluded that the failure neutralization is much better in AntOR-DLR.

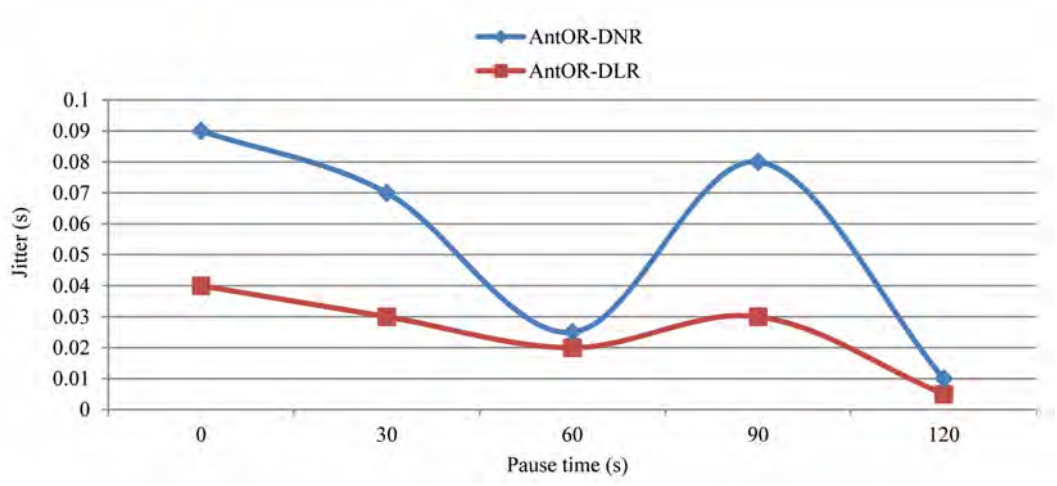


Figure 7.8: Jitter (AntOR-DNR)

7.6 Evaluation of AntOR-RDLR Protocol

To evaluate the performance of the AntOR-RDLR protocol, in terms of effectiveness, the impact of the increase of the pause time has been taken into account, that is, how this affects parameters such as the throughput and the delivered data packet ratio. This evaluation was developed jointly with the AntOR-DLR protocol . Previously we analysis in this subsection which values for the MAX_HOP parameter from AntOR-RDLR protocol are suitable.

7.6.1 Setting of MAX_HOP

As shown in Figure 7.9, the optimal value of MAX_HOP, in terms of delivered data packet ratio, it is reached for a value of 6.

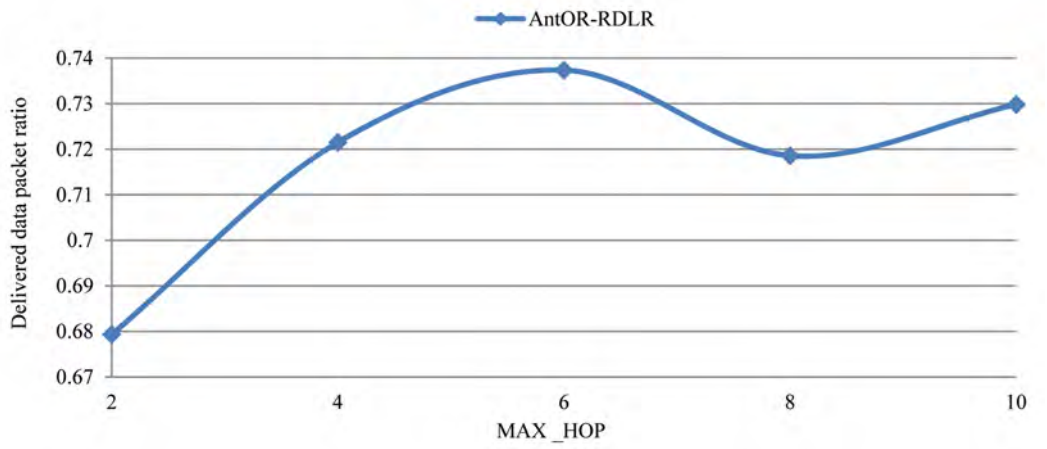


Figure 7.9: Setting of MAX_HOP - case a (AntOR-RDLR)

On the other hand, and as shown in Figure 7.10, the optimal value of MAX_HOP, in

terms of overhead in the number of bytes, it is reached for the minimum value (2 in this case).

Consequently, in view of the two previous graphs the best benefits of AntOR-RDLR are achieved for values of MAX_HOP in the interval [2, 6]. In the comparison that follows a value of 5 for MAX_HOP was chosen.

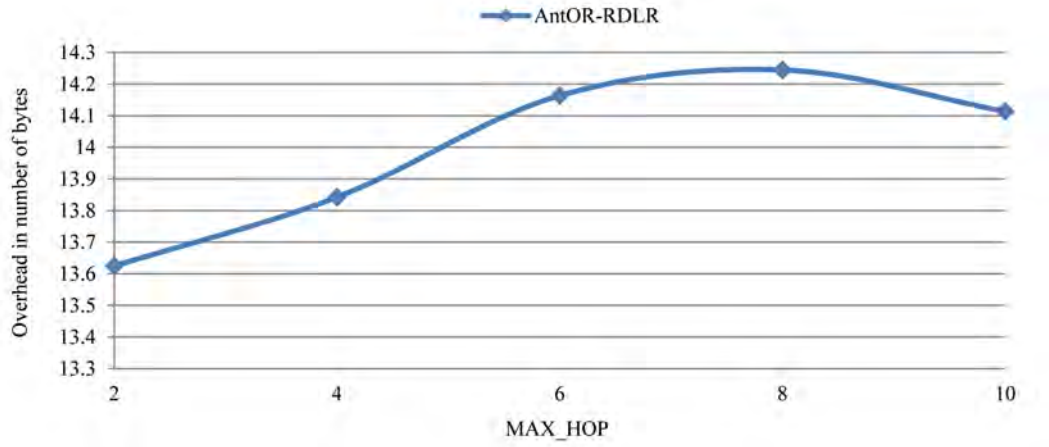


Figure 7.10: Setting of MAX_HOP - case b (AntOR-RDLR)

7.6.2 Throughput

As shown in Figure 7.11, the throughput in AntOR-RDLR is, at all times, higher than its predecessor, AntOR-DLR. This improvement in the throughput is a consequence of the greater fault tolerance of AntOR-RDLR, due to the possibility of having more alternative routes using nodes that belong to the main route, which does not occur in AntOR-DLR.

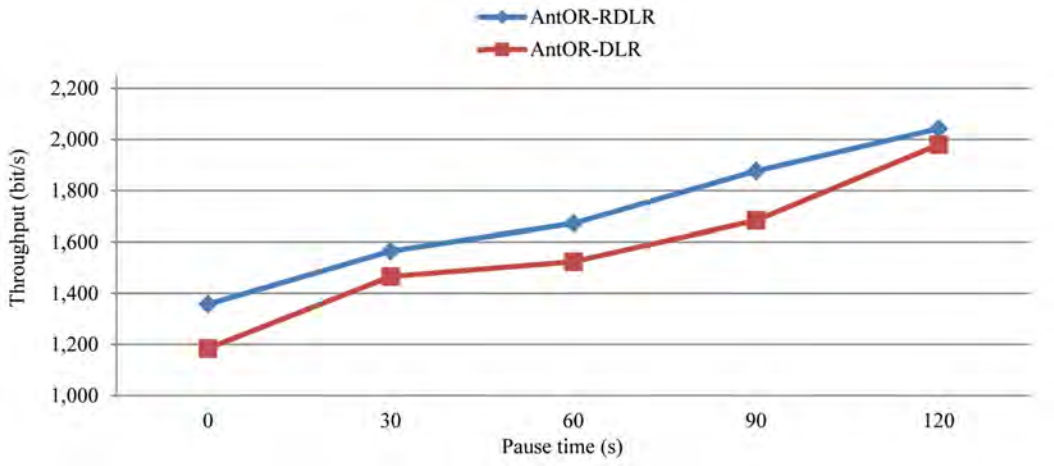


Figure 7.11: Throughput (AntOR-RDLR)

7.6.3 Delivered Data Packet Ratio

As shown in Figure 7.12, the delivered rata packet ratio in AntOR-RDLR is, at all times, higher than its predecessor. The explanation of this fact is similar to the one carried out in the previous subsection.

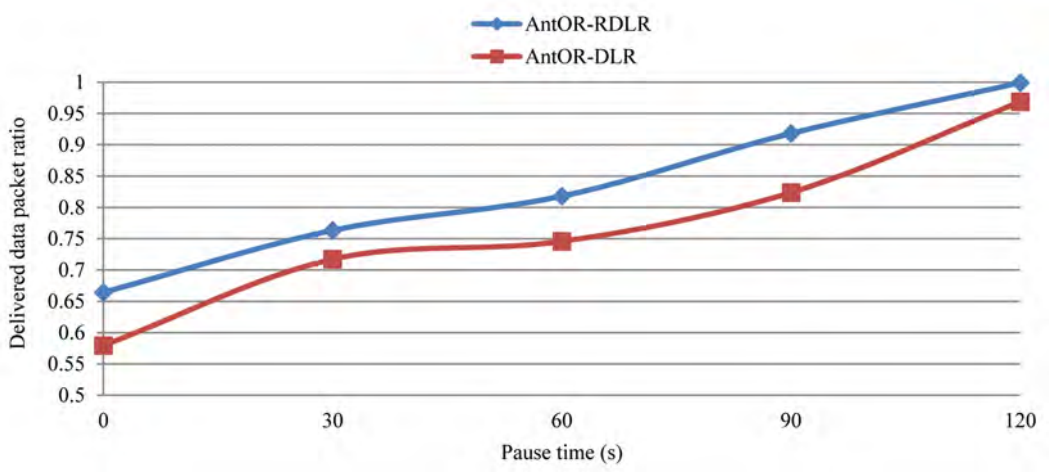


Figure 7.12: Delivered data packet ratio (AntOR-RDLR)

7.7 Evaluation of AntOR-UDLR Protocol

To evaluate the benefits of AntOR-UDLR protocol, both in terms of efficiency and effectiveness, the impact of the increase of the pause time has been taken into account and how this affects parameters such as the throughput, the delivered data packet ratio, average end-to-end delay and the overhead in number of packets. Also, it took into account the impact of the increase of the speed of the nodes and how this affects parameters such as the throughput, the delivered data packet ratio, average end-to-end delay and the overhead in number of bytes. This evaluation was developed jointly with the AntOR-DLR and OLSR protocols.

7.7.1 Throughput

As shown in Figure 7.13, the throughput in AntOR-UDLR is, at all times, higher than its predecessor, AntOR-DLR, regardless of the pause time.

Similarly, as shown in Figure 7.14, the throughput in AntOR-UDLR is also, at all times, higher than its predecessor, AntOR-DLR, regardless of the speed of the nodes.

From the above we can conclude that presented modifications in AntOR-ULDR improve the effectiveness of the protocol. In other words, the process of link failure neutralization is faster in AntOR-UDLR by sending unicast packets, more reliable than broadcast packets utilized by its predecessor.

In addition, it should be point out the difference between AntOR-UDLR / AntOR-DLR with respect to OLSR, difference that expands considerably in very dynamic scenarios, which is explained by the proactive nature of this latter easily.

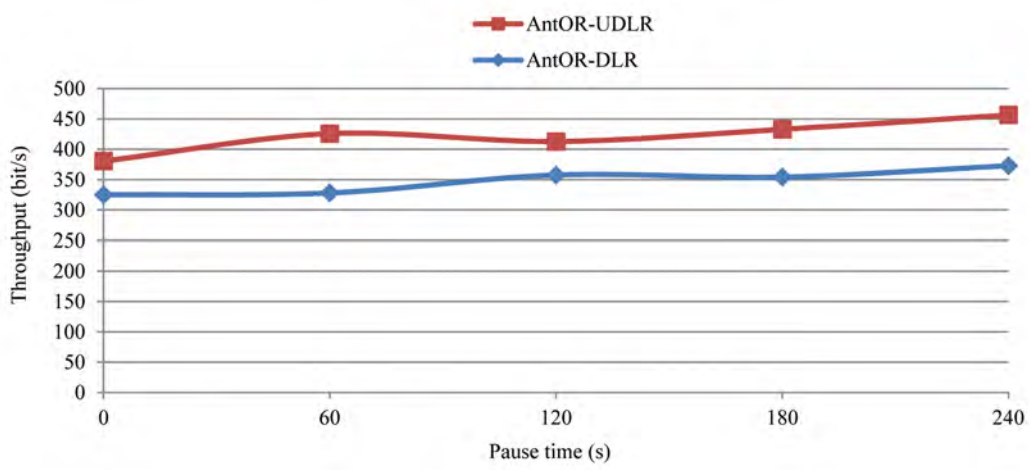


Figure 7.13: Throughput - case a (AntOR-UDLR)

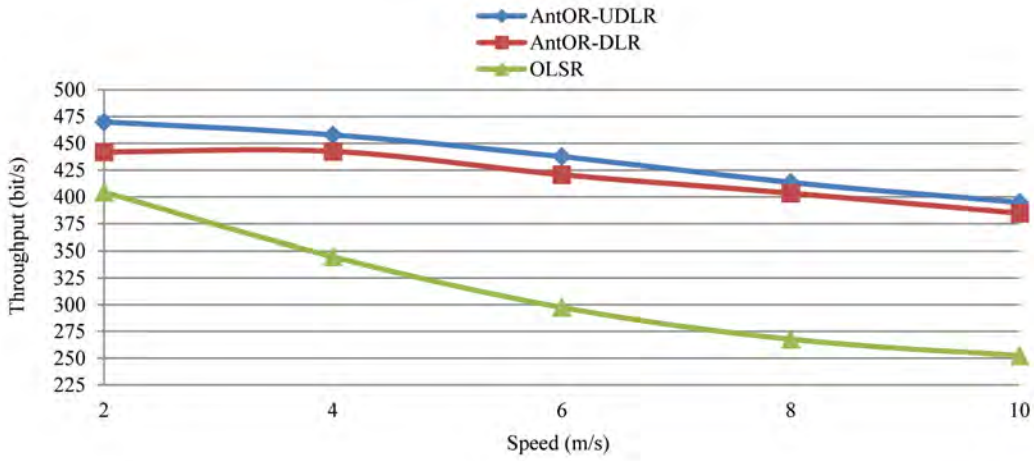


Figure 7.14: Throughput - case b (AntOR-UDLR)

7.7.2 Delivered Data Packet Ratio

As shown in Figures 7.15 and 7.16, the Delivered Data Packet Ratio in AntOR-UDLR is, at all times, higher than AntOR-DLR. Also, both AntOR-UDLR and AntOR-DLR enhance to OLSR. The explanation of this fact is similar to the one carried out in the previous subsection.

7.7.3 Average End-to-End Delay

As shown in Figures 7.17 and 7.18, the average end-to-end delay in AntOR-UDLR is, at all times, less than AntOR-DLR, besides being more uniform. This last fact allows it to prove the good properties of scalability in AntOR-UDLR. Likewise, we note how delay in OLSR is still lower than in both protocols. This is because OLSR, by being purely proactive, presents a low latency. It should be recalled that the protocols designed in this Thesis are hybrid, being these differences in normal values that are found in the literature.

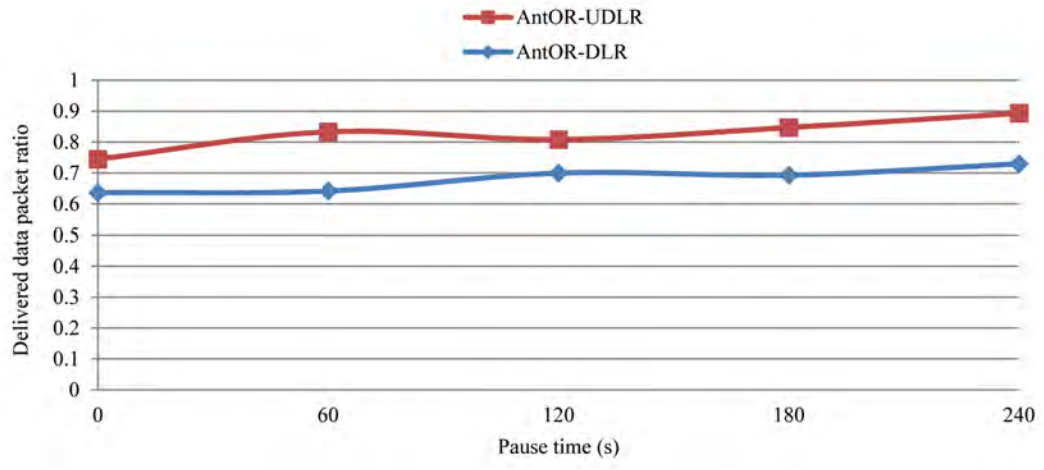


Figure 7.15: Delivered data packet ratio - case a (AntOR-UDLR)

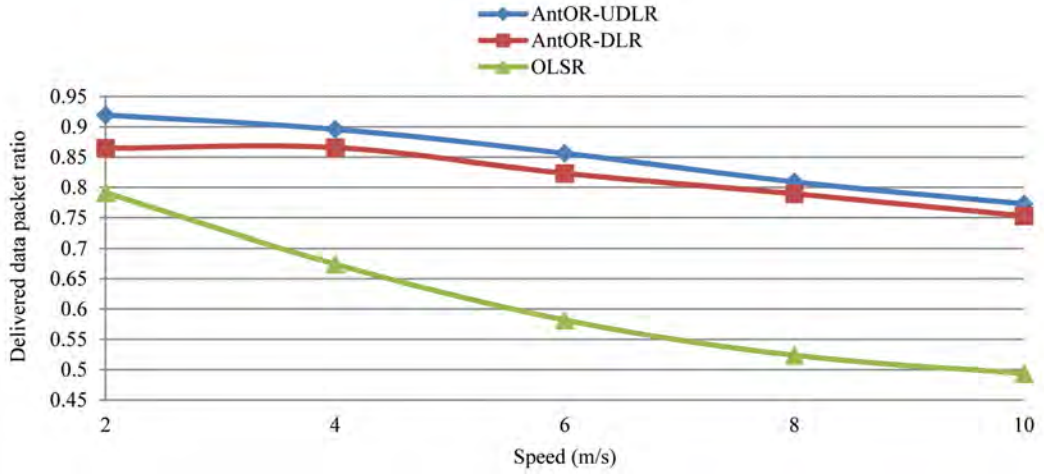


Figure 7.16: Delivered data packet ratio - case b (AntOR-UDLR)

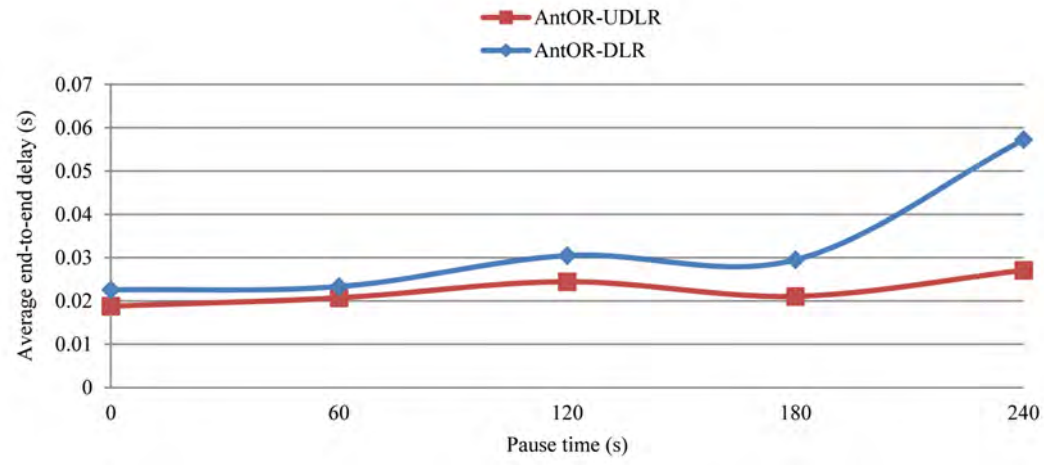


Figure 7.17: Average end-to-end delay - case a (AntOR-UDLR)

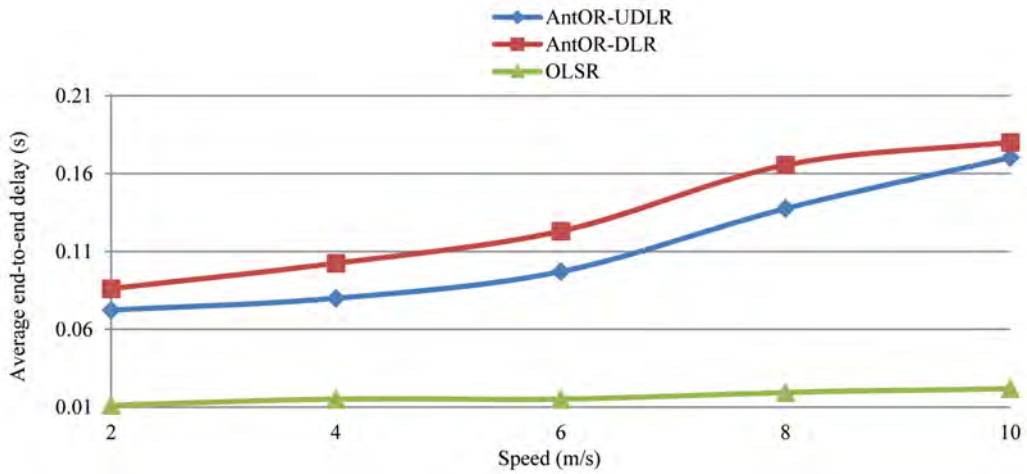


Figure 7.18: Average end-to-end delay - case b (AntOR-UDLR)

7.7.4 Overhead in the Number of Packets

As shown in Figure 7.19, the overhead in the number of packets in AntOR-UDLR is, in general terms, similar to AntOR-DLR.

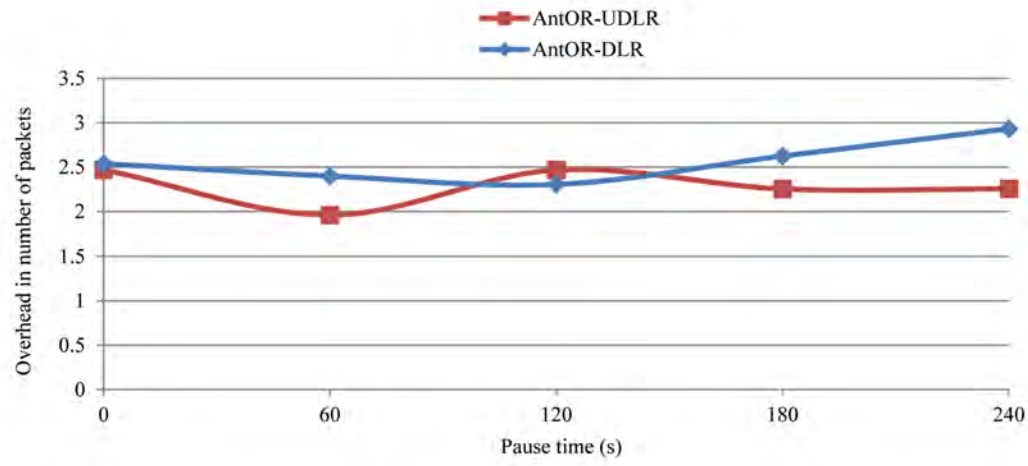


Figure 7.19: Overhead in the Number of Packets (AntOR-UDLR)

7.7.5 Overhead in the Number of Bytes

As shown in Figure 7.20, and identically to what is mentioned in the preceding subsection, the overhead in the number of bytes in AntOR-UDLR is, in general terms similar to AntOR-DLR. Likewise, it notes that these overloads are significantly lower than the OLSR, which is logical given the proactive character of this.

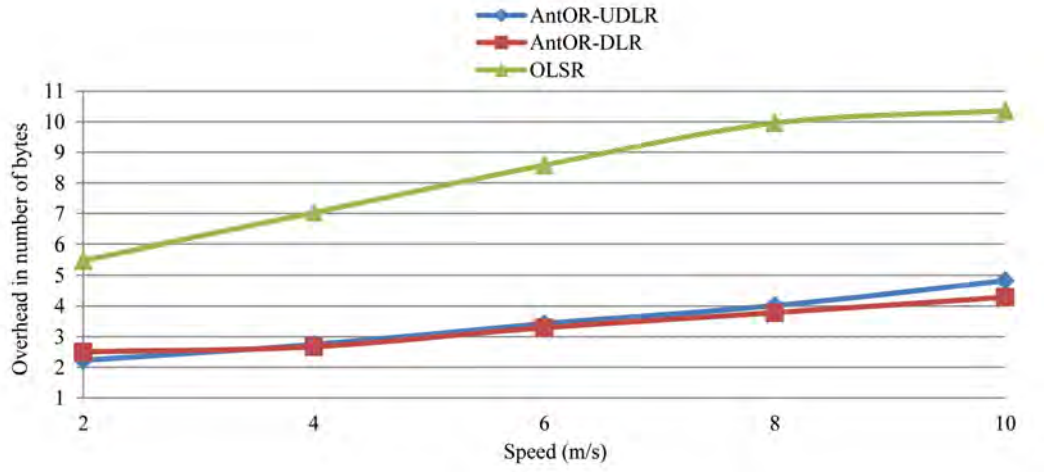


Figure 7.20: Overhead in the Number of Bytes (AntOR-UDLR)

7.8 Evaluation of AntOR-v2 Protocol

To assess benefits of AntOR-v2 protocol, both in terms of efficiency and effectiveness, we have taken into account the impact of the increase in the pause time and how this affects parameters such as delivered data packet ratio, Jitter, overhead in the number of packets. Also, the impact of the increase in the number of nodes has been taken into account and how this affects parameters such as the throughput, the delivered data packet ratio, average end-to-end delay, jitter, overhead in the number of bytes. This evaluation was developed jointly with the AODV protocol. AODV was chosen for two reasons: first of all, the majority of hybrid protocols are compared in the literature with its (and it is a required reference); Secondly, a comparison with a reactive protocol such as AODV has been chosen for this protocol since previously, the comparison had been made with a proactive protocol such as OLSR.

7.8.1 Throughput

As shown in Figure 7.21, the throughput in AntOR-v2 is, at all times, greater than the AODV, regardless of the number of nodes, being especially significant difference in dense networks. Also, the throughput in AntOR-v2 decays slowly in such networks. Both facts determine a good behavior of AntOR-v2 with respect to the scalability of the network.

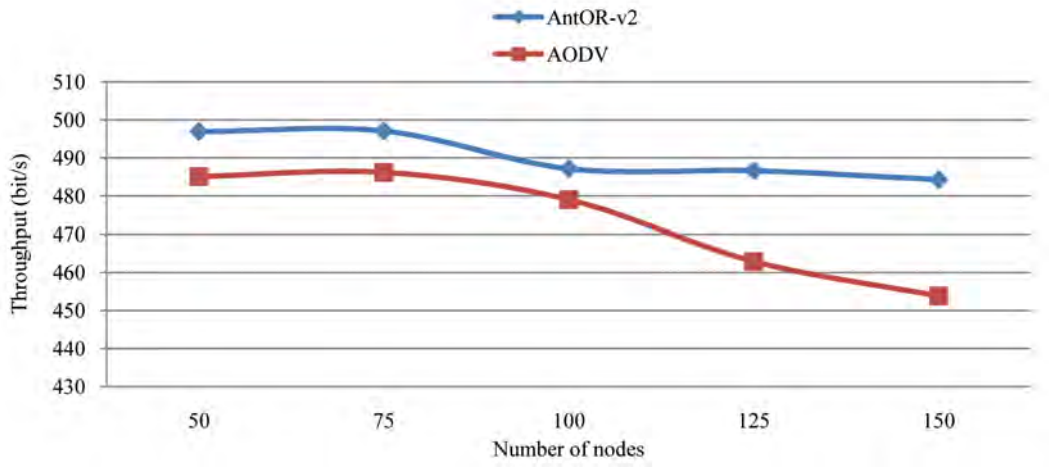


Figure 7.21: Throughput (AntOR-v2)

7.8.2 Delivered Data Packet Ratio

As shown in the Figures 7.22 and 7.23, and similarly to what has been mentioned in the previous subsection, the delivered data packet ratio in AntOR-v2 is, at all times, greater than the AODV, regardless of the number of nodes, being especially significant difference between them in dense networks. In addition, the Delivered Data Packet Ratio in AntOR-v2 decays into such networks slowly. Both facts determine a good behavior of AntOR-v2 with respect to the scalability of the network.

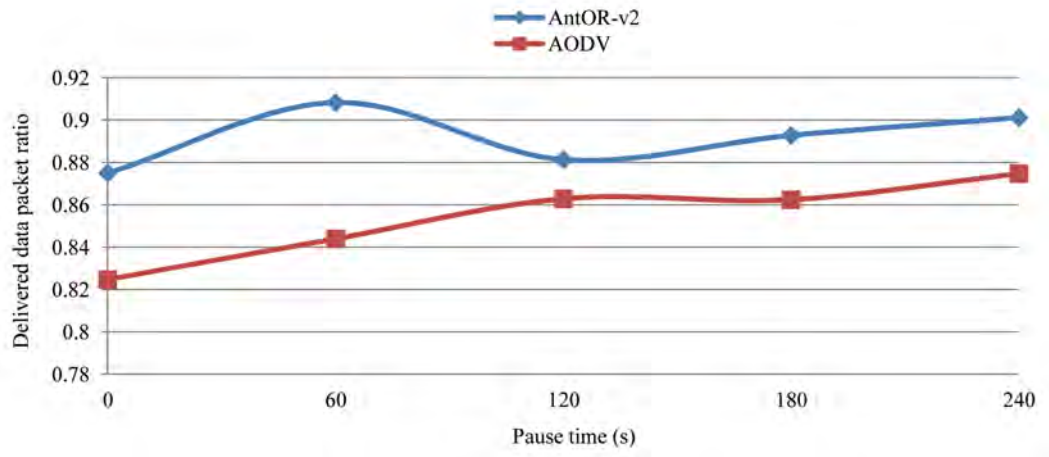


Figure 7.22: Delivered data packet ratio - case a (AntOR-v2)

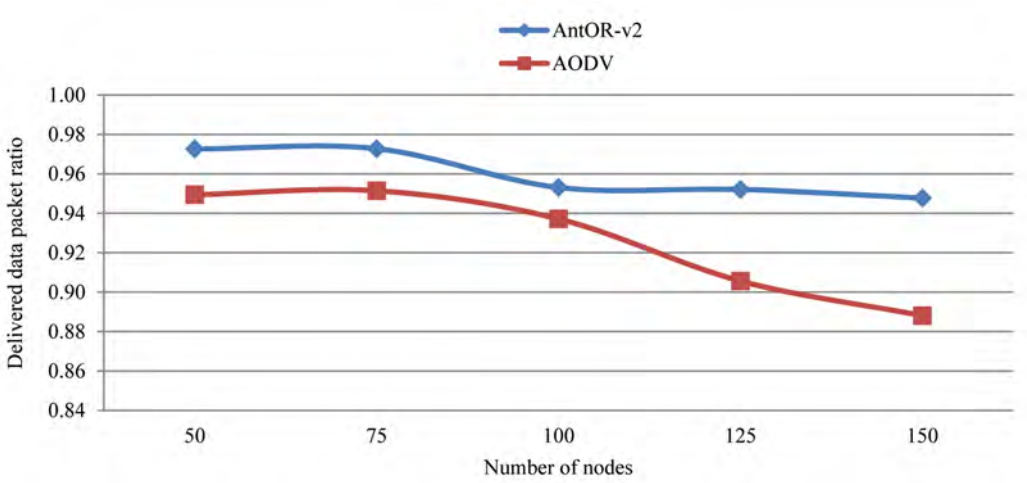


Figure 7.23: Delivered data packet ratio - case b (AntOR-v2)

7.8.3 Average End-to-End Delay

As shown in Figure 7.24, average end-to-end delay in AntOR-v2 is, at all times, less than AODV. This is logical given the reactive nature of AODV.

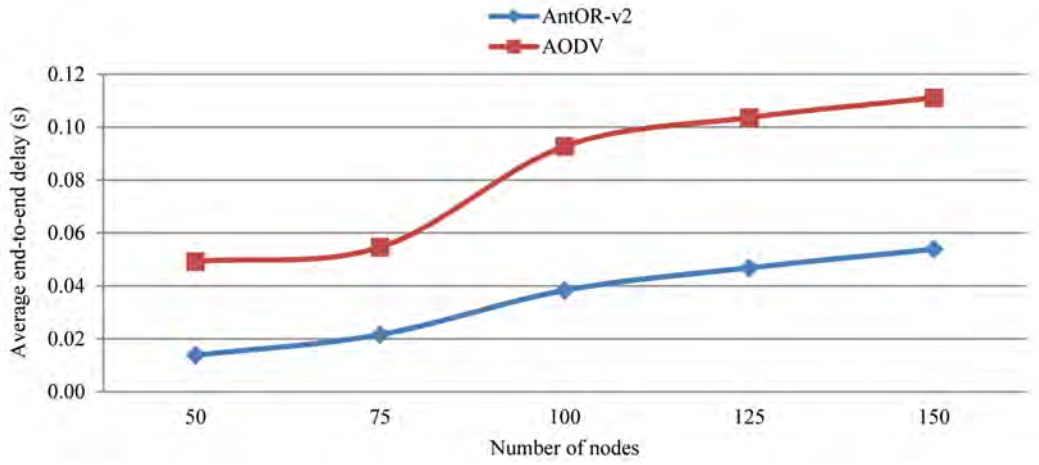


Figure 7.24: Average end-to-end delay (AntOR-v2)

7.8.4 Jitter

As shown in Figure 7.25, the jitter at AntOR-v2 is in terms generals, inferior to the AODV and practically constant, regardless of the pause time, making AntOR-v2 to be a fairly robust protocol. In addition, and as shown in Figure 7.26, el *jitter* the jitter in AntOR-v2 is, at all times, clearly inferior than AODV, regardless of the number of nodes, presenting a similar monotonous behavior.

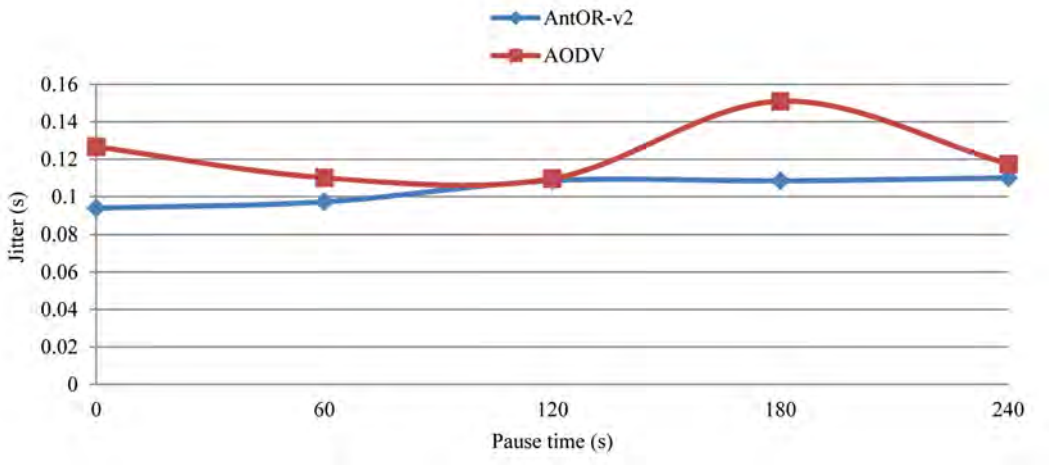


Figure 7.25: Jitter - case a (AntOR-v2)

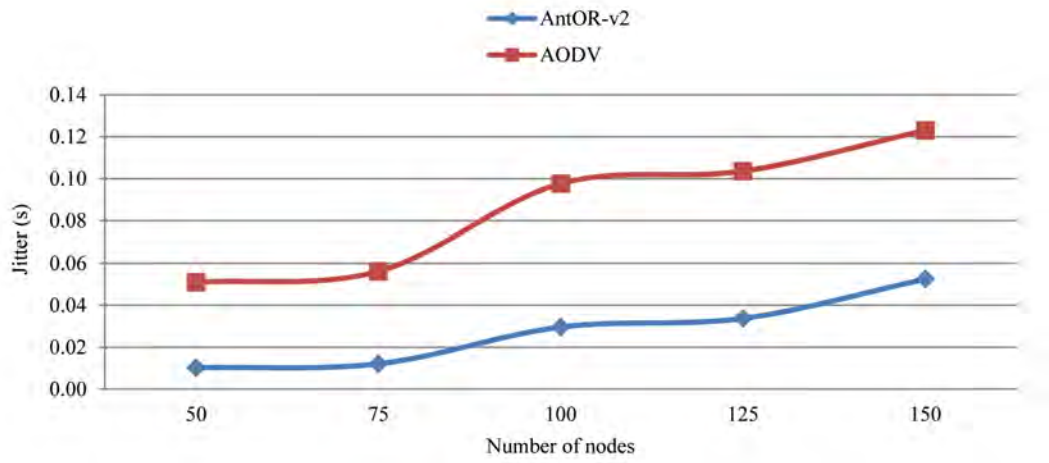


Figure 7.26: Jitter - case b (AntOR-v2)

7.8.5 Overhead in Number of Packets

As shown in the figure 7.27, Overhead in number of packets in AntOR-v2 is very similar to the AODV, regardless of the pause time. On the other hand, and as shown in the Figure 7.28, the overhead in AntOR-v2 is, at all times, slightly higher than AODV, regardless of the number of nodes, narrowing the difference in dense networks.

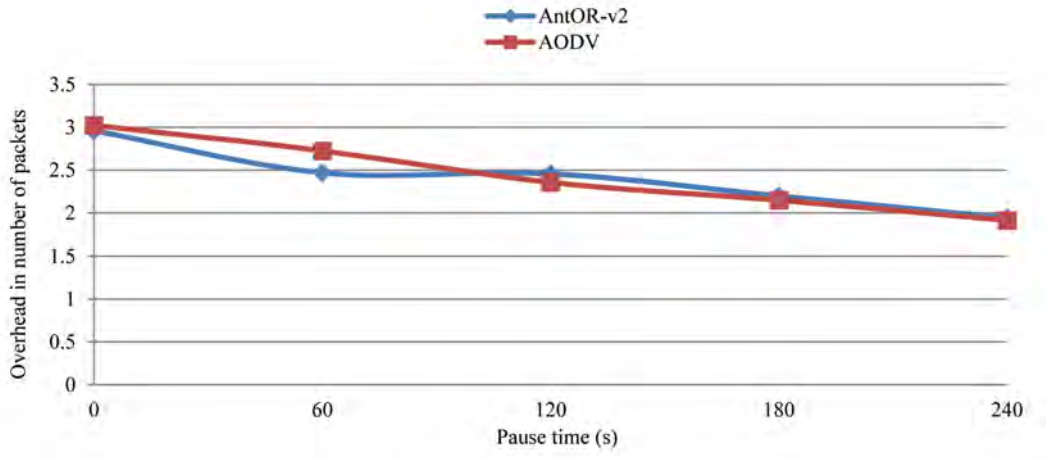


Figure 7.27: Overhead in number of packets - case a (AntOR-v2)

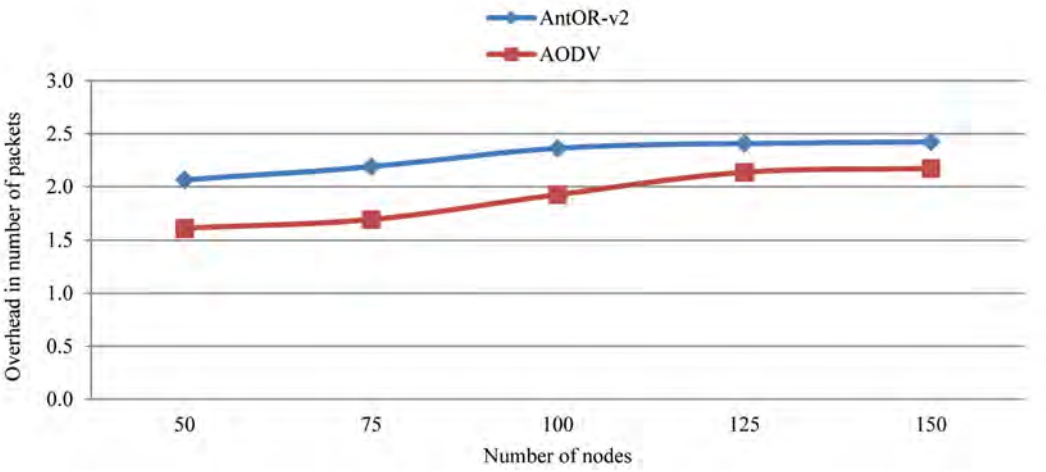


Figure 7.28: Overhead in number of packets - case b (AntOR-v2)

7.8.6 Overhead in Number of Bytes

As shown in Figure 7.29, and similarly to what mentioned in Figure 7.28, the overhead in AntOR-v2 is, at all times, slightly higher than AODV, regardless of the number of nodes, narrowing the difference in dense networks.

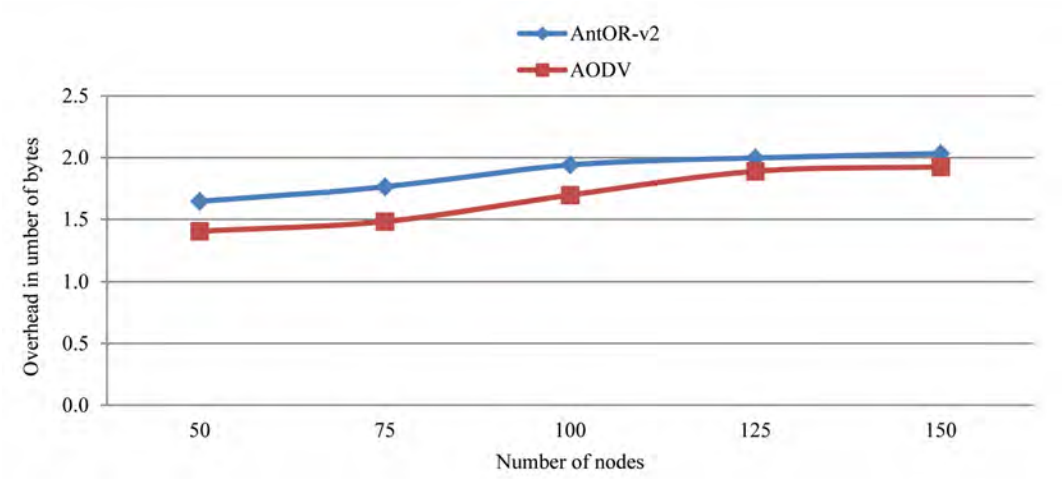


Figure 7.29: Overhead in number of bytes (AntOR-v2)

7.9 Evaluation of HACOR Protocol

To evaluate the performance of the HACOR protocol, both in terms of efficiency and effectiveness, has been taken into account the impact of the increase of the pause time and how this affects parameters such as the throughput, delivered data packet ratio, average end-to-end Delay, Jitter, overhead in number of packets and overhead in the number of bytes. Also, it is has taken into account the impact of the increase in the number of nodes and how this affects parameters such as the throughput, delivered data packet ratio, average end-to-end delay, jitter, overhead in the number of packets and overhead in the number of bytes. This evaluation has been carried out jointly with the standards AODV and OLSR.

7.9.1 Throughput

As shown in Figures 7.30 and 7.31, the throughput in HACOR is, at all times, greater than the other two protocols. Also, it hardly decays with the number of nodes, which allows it to conclude its good predisposition for scalability.

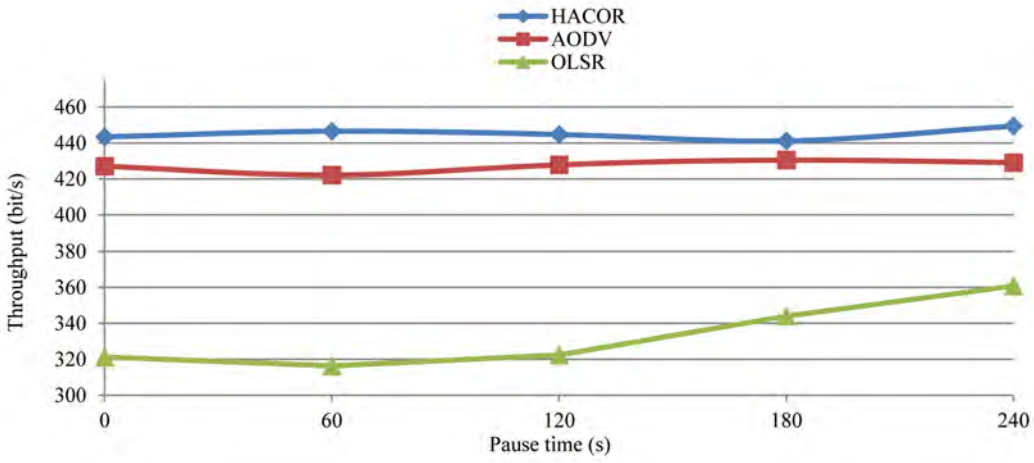


Figure 7.30: Throughput - case a (HACOR)

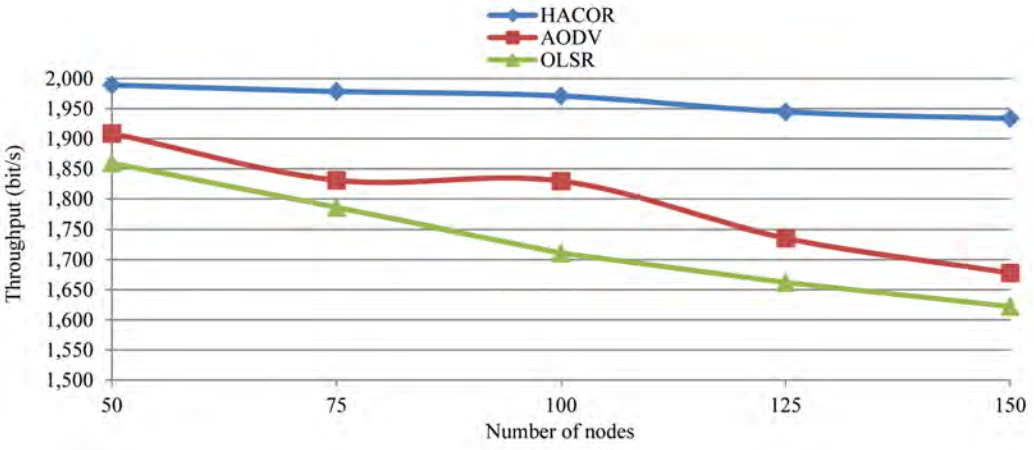


Figure 7.31: Throughput - case b (HACOR)

7.9.2 Delivered Data Packet Ratio

As shown in Figures 7.32 and 7.33, and similarly mentioned in the previous paragraph, the delivered data packet ratio in HACOR is, at all times, greater than the of the other two protocols. Also, it barely decays with the number of nodes, allowing it to reaffirm your good predisposition for scalability.

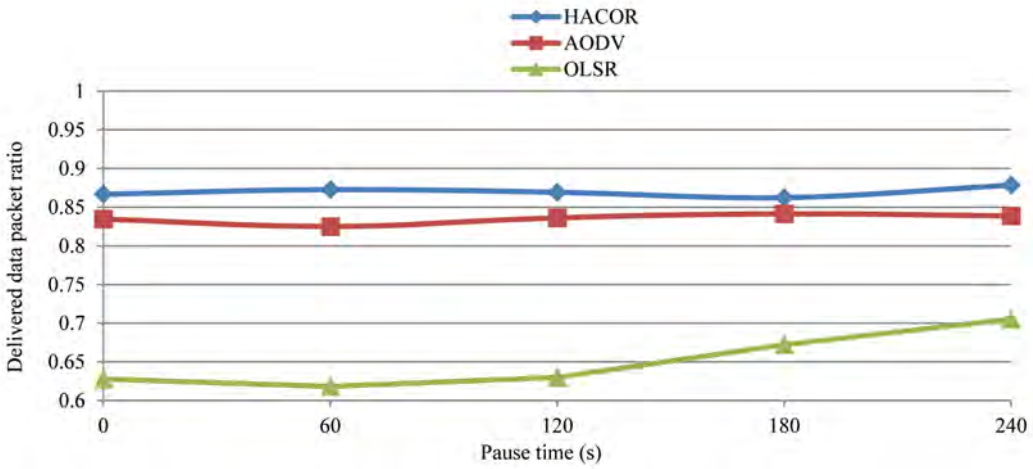


Figure 7.32: Delivered data packet ratio - case a (HACOR)

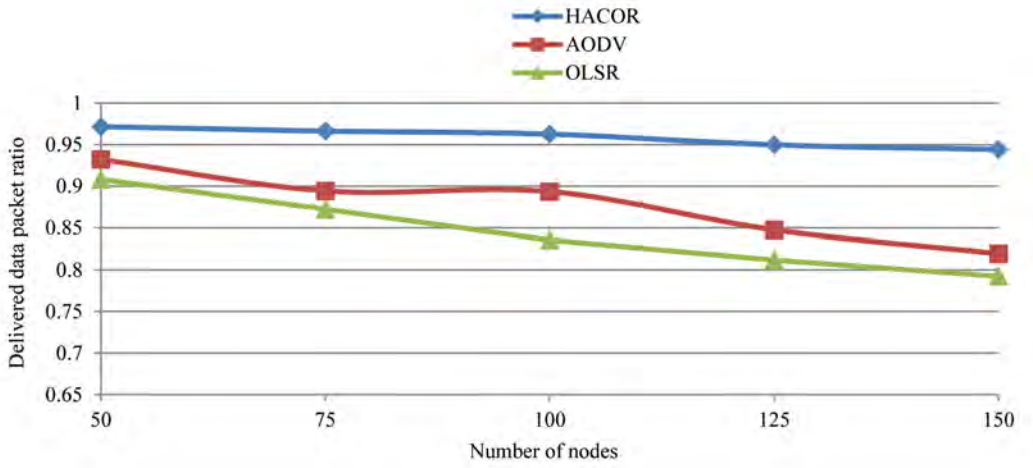


Figure 7.33: Delivered data packet ratio - case b (HACOR)

7.9.3 Average End-to-End Delay

As shown in Figures 7.34 and 7.35, the average end-to-end delay in HACOR takes intermediate values to AODV and OLSR. This is logical since the latency of a hybrid protocol usually ranges between a reactive one and a proactive one. However, and as shown in Figure 7.35, delay is in HACOR slightly higher than OLSR, regardless of the number of nodes. This is especially noteworthy since HACOR minimizes considerably Average End-to-End Delay, showing its proactive character and hiding, in certain way, its reactive nature. As in other two metrics analyzed above we can conclude its excellent predisposition for scalability.

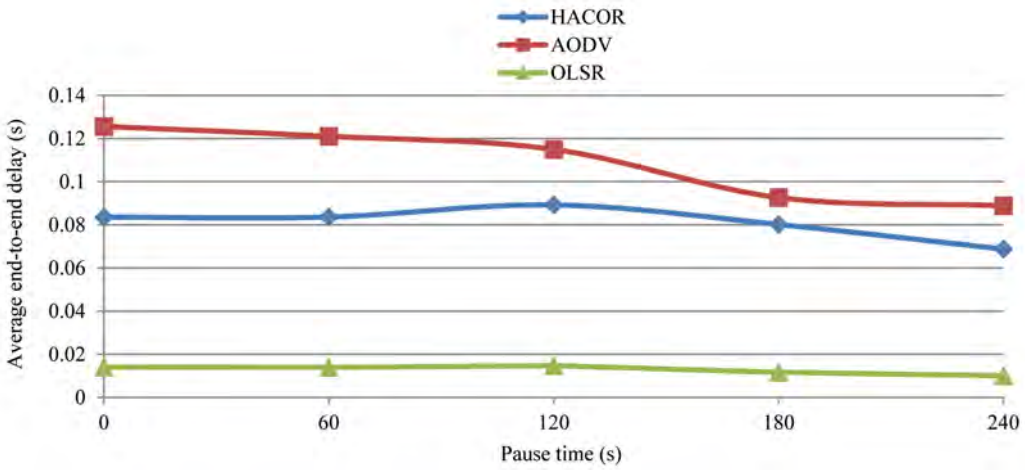


Figure 7.34: Average end-to-end delay - case a (HACOR)

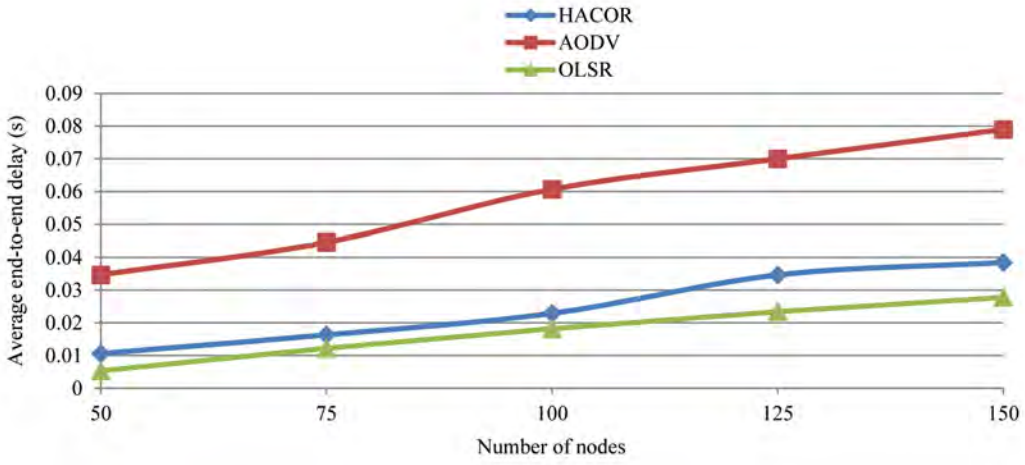


Figure 7.35: Average end-to-end delay - case b (HACOR)

7.9.4 Jitter

As shown in Figure 7.36, and in a similar way to what has been mentioned in the preceding subsection, the jitter in HACOR takes intermediate values to the AODV and OLSR regarding the pause time. In addition, and as shown in Figure 7.37, HACOR is the most robust presenting similar values of jitter, regardless of the number of nodes, improving from a threshold in the number of nodes to AODV, circumstance which is accentuated, particularly dense networks. As in the metrics discussed above, we can conclude its excellent predisposition for scalability.

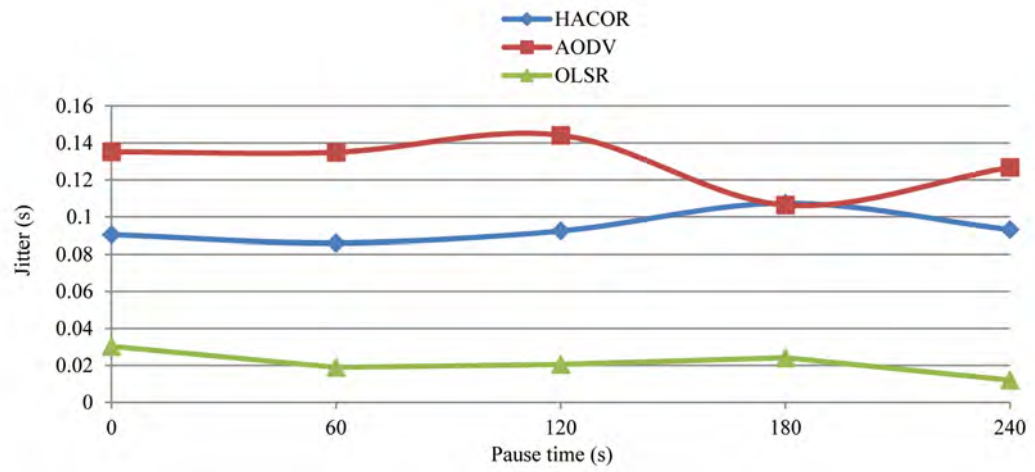


Figure 7.36: Jitter - case a (HACOR)

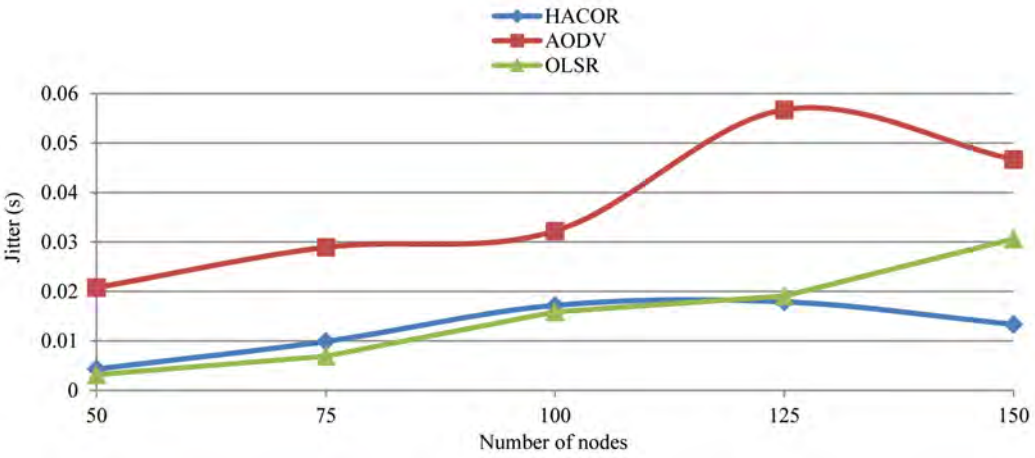


Figure 7.37: Jitter - case b (HACOR)

7.9.5 Overhead in Number of Packets

As shown in Figure 7.38, the overhead in number of packets in HACOR take intermediate values to AODV and OLSR regarding the pause time. However, and as shown in Figure 7.39, overhead is higher than that of AODV and OLSR with respect to the number of nodes. It should be mentioned as favorable aspect that in dense networks the differences with OLSR are narrowed to become null or almost null. It can conclude that the price to pay for the metrics discussed above is a slight increase in overload, price that decreases as increases the number of nodes, which also somehow affirms its good predisposition for scalability.

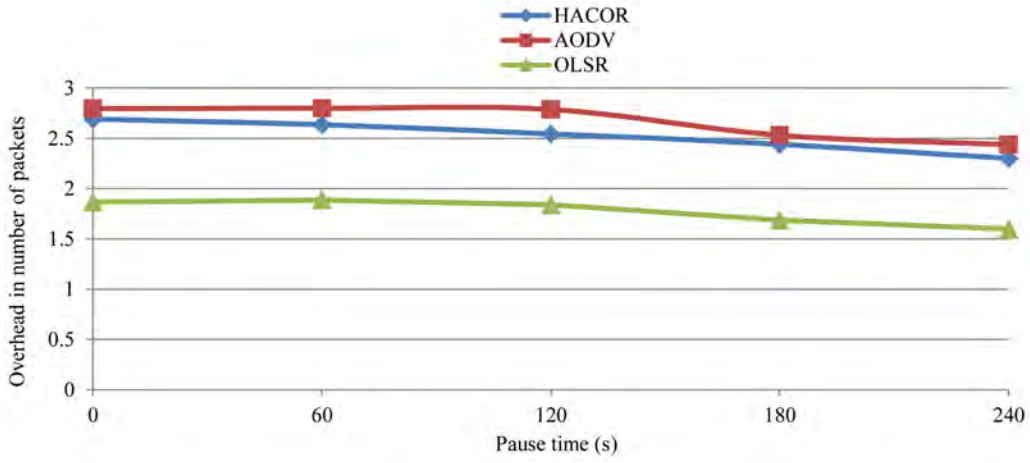


Figure 7.38: Overhead in number of packets - case a (HACOR)

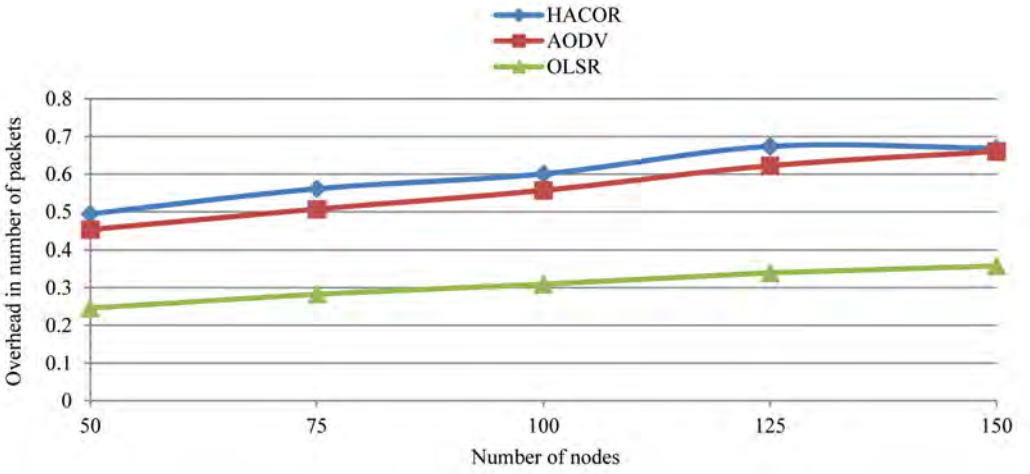


Figure 7.39: Overhead in number of packets - case b (HACOR)

7.9.6 Overhead in Number of Bytes

As shown in Figures 7.40 and 7.41, the overhead in number of bytes in AntOR has a behavior similar to the Overhead in Number of pPckets, being able to conclude the indicated previously.

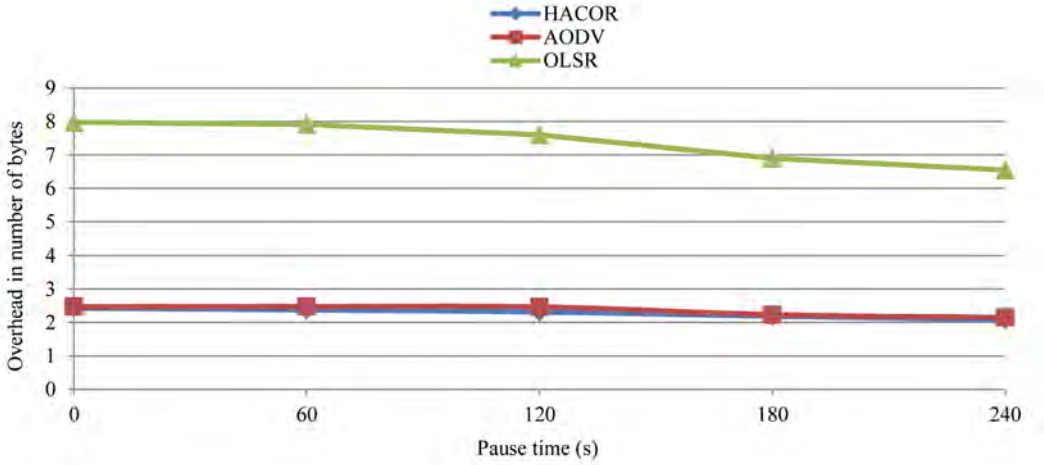


Figure 7.40: Overhead in number of bytes - case a (HACOR)

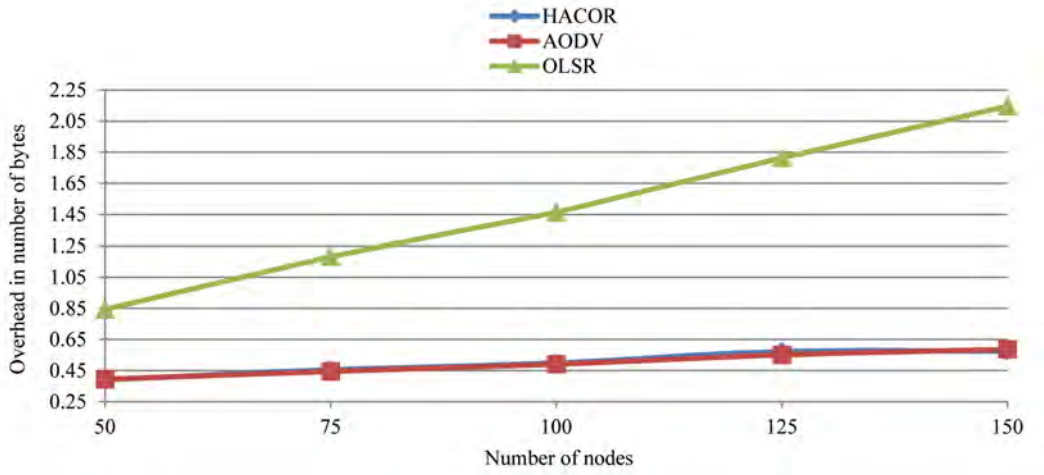


Figure 7.41: Overhead in number of bytes - case b (HACOR)

7.10 Evaluation of PAntOR Protocol

To evaluate the benefits of the PAntOR protocol, both in terms of efficiency and effectiveness, the impact of the increase of the speed of the nodes has been taken into account and how this affects parameters such as the throughput, the delivered data packet ratio, average end-to-End delay, jitter and overhead in number of packets. Also, the impact of the increase of the pause time has been taken into account and how this affects parameters such as the delivered data packet ratio, average end-to-end delay and jitter. This evaluation was developed jointly with its predecessor protocol, AntOR-DNR.

7.10.1 Throughput

As shown in Figure 7.42, the throughput in PAntOR is higher, at all times, than AntOR-DNR. This is easily explained by the parallelization introduced in the route setup

phase and in the route local repair processes and link failure notification.

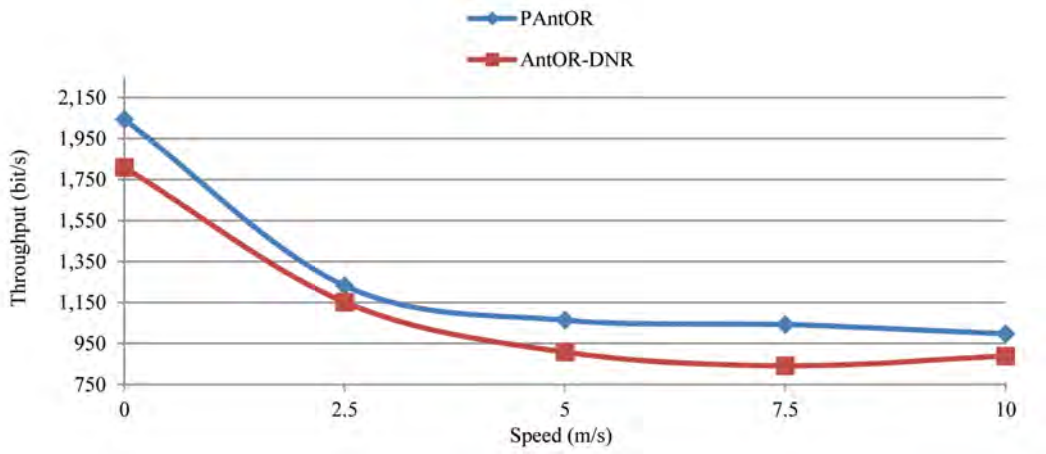


Figure 7.42: Throughput (PAntOR)

7.10.2 Delivered Data Packet Ratio

As shown in Figures 7.43 and 7.44, the delivered data packet ratio in PAntOR is higher, at all times, than AntOR-DNR. As noted above, this is easily explained by the parallelization introduced in the route setup phase and in the route local repair processes and link failure notification.

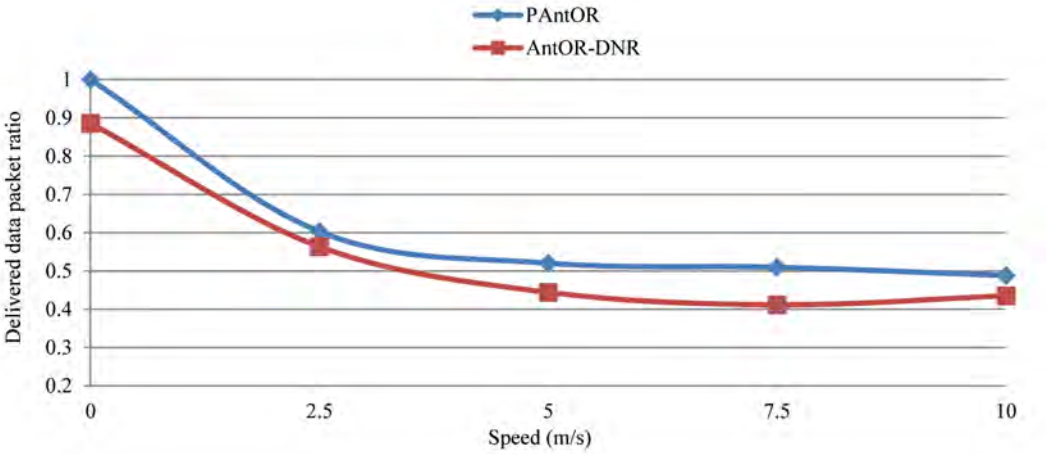


Figure 7.43: Delivered data packet ratio - case a (PAntOR)

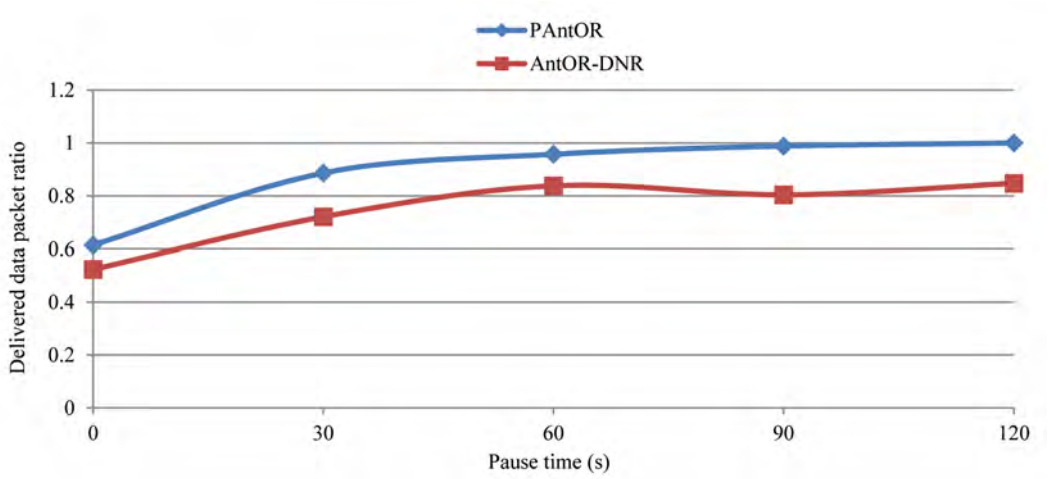


Figure 7.44: Delivered data packet ratio - case b (PAntOR)

7.10.3 Average End-to-End Delay

As shown in Figures 7.45 and 7.46, the average end-to-end delay is, at all times, less than its predecessor. This improvement is the result of the parallelization made, especially of what is introduced in the route setup phase (phase 1 of the protocol).

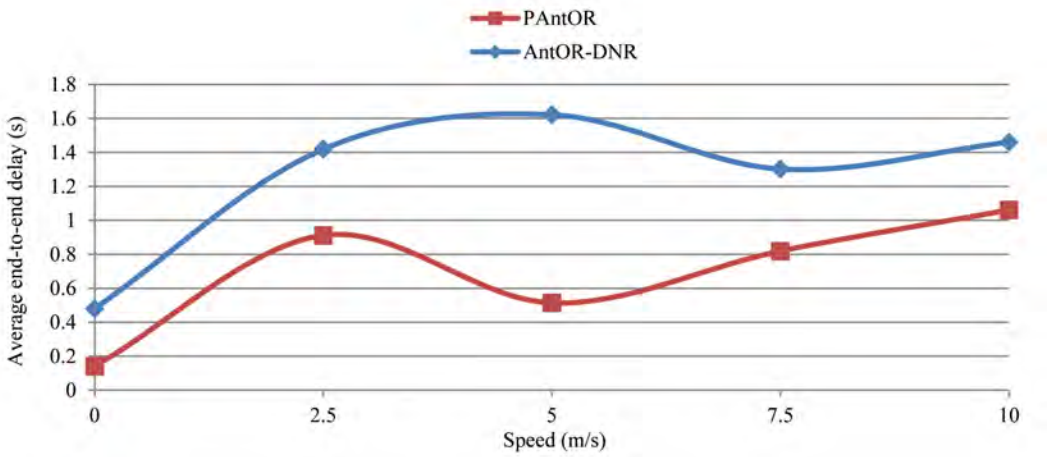


Figure 7.45: Average end-to-end delay - case a (PAntOR)

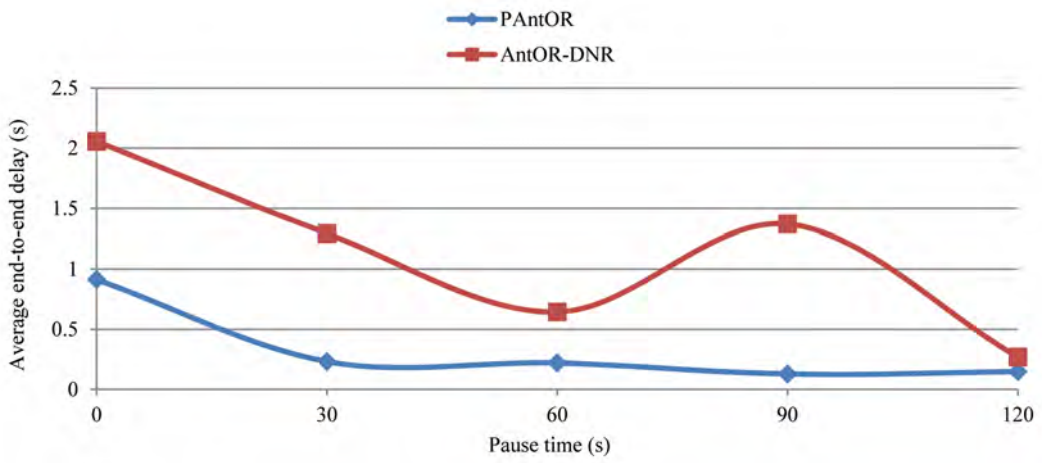


Figure 7.46: Average end-to-end delay - case b (PAntOR)

7.10.4 Jitter

As shown in Figures 7.47 and 7.48, the jitter in PAntOR is lower, at all times, than AntOR-DNR. This improvement is the result of parallelization made, especially of what is introduced in the local route repair processes and link failure notification (phase 4 of the protocol).

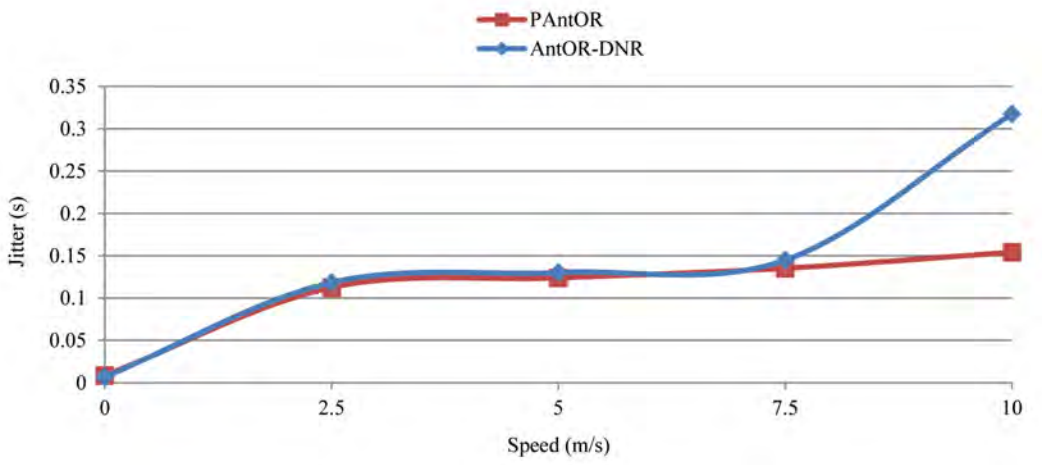


Figure 7.47: Jitter - case a (PAntOR)

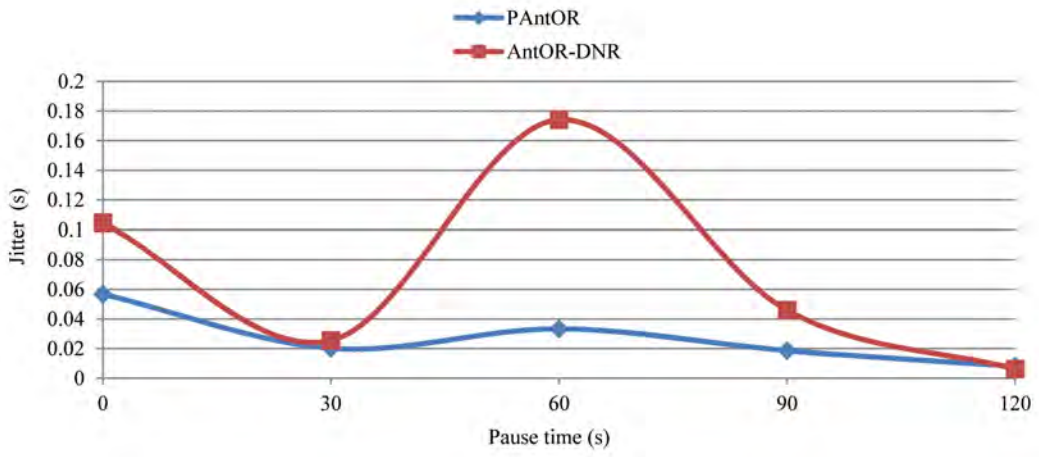


Figure 7.48: Jitter - case b (PAntOR)

7.10.5 Overhead in Number of Packets

As shown in Figure 7.49, the overhead in number of packets in PAntOR is, at all times, lower than its predecessor. Likewise it notes how this difference increases with the speed of the nodes (or link break). This is easily explained because P-AntOR speeds up the route local repair processes, also avoiding the broadcast mode since the agents are sent to the neighbors of the local node which perceives the failure.

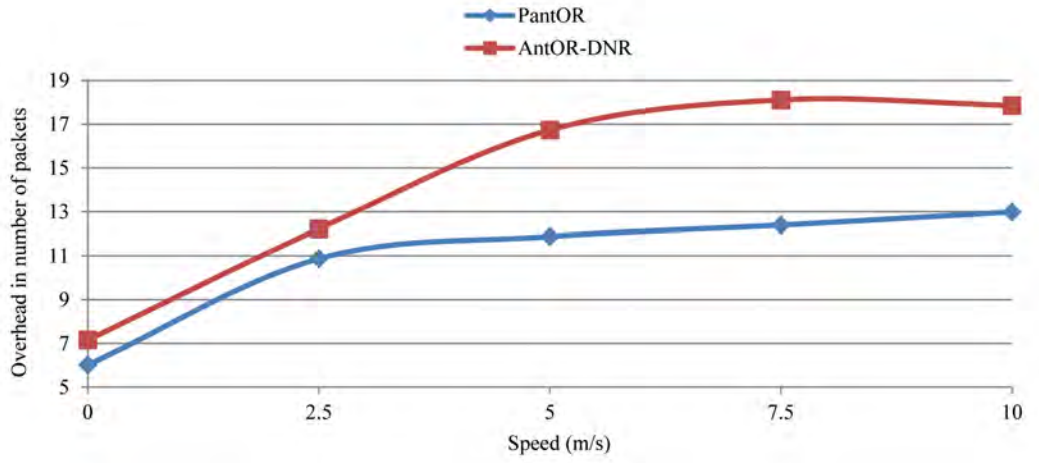


Figure 7.49: Overhead in number of packets (PAntOR)

7.11 Evaluation of PAntOR-MI Protocol

To evaluate the benefits of the protocol PAntOR-MI, in terms of effectiveness, the impact of the increase of the speed of the nodes has been taken into account and how this affects the delivered data packet ratio. This evaluation was developed jointly with the protocols P-AntOR and AntOR-DNR.

7.11.1 Delivered Data Packet Ratio

As shown in Figure 7.50, the delivered data packet ratio in PAntOR-MI is better than its predecessor in dynamic environments. This is easily explained because PAntOR-MI parallelizes the route setup phase (phase 1 of the protocol) through threads with the usage of the multi-interface, losing fewer data packets.

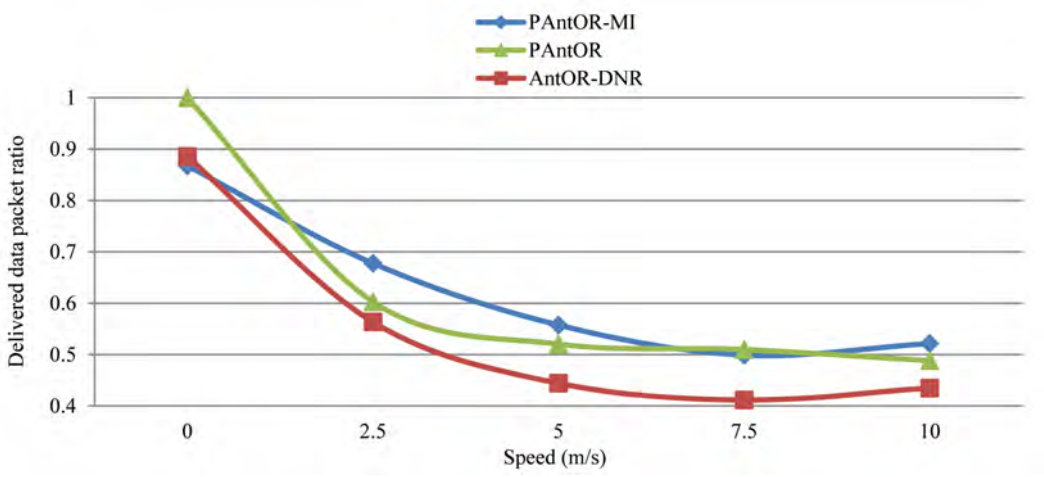


Figure 7.50: Delivered data packet ratio (PAntOR-MI)

7.12 Summary

This chapter has collected the simulations carried out considering real scenarios in order to verify the applicability of AntOR, AntOR-DLR, AntOR-DNR, AntOR-RDLR, AntOR-UDLR, AntOR-v2, HACOR, PAntOR and PAntOR-MI. To this end, we have utilized the Network Simulator NS-3 [NS3].

In AntOR-DLR we have evaluated the following metrics: throughput, delivered data packet ratio, average end-to-end delay, overhead in the number of packets and overhead in the number of bytes. The simulation results demonstrate protocol AntOR-DLR presents respect to AntHocNet better throughput and delivered data packet ratio as well as a small increase in the average end-to-end delay and overhead, negative aspects these last ones that are equal in dense networks.

In AntOR-DNR we have evaluated the following metrics: delivered data packet ratio, average end-to-end delay and jitter. The results of the simulations show that protocol AntOR-DLR improves to AntOR-DNR in all the analyzed metrics.

AntOR-RDLR has evaluated the following metrics: throughput and delivered data packet ratio. The simulation results show that protocol AntOR-RDLR improves to AntOR-DLR in all the analyzed metrics.

In AntOR-UDLR we have evaluated the following metrics: throughput, delivered data packet ratio, average end-to-end delay, overhead in the number of packets and overhead in the number of bytes. The simulation results show that protocol AntOR-UDLR improves to its predecessor in all the analyzed metrics, with the exception of the overhead, in which case both protocols present similar values.

In AntOR-v2 we have evaluated the following metrics: throughput, delivered data packet ratio, average end-to-end delay, jitter, overhead in the number of packets and overhead in the number of bytes. The simulation results show that AntOR-v2 Protocol

improves to AODV in all the analyzed metrics, with the exception of the overhead that is slightly higher, difference is that is imperceptible in dense networks.

In HACOR we have evaluated the following metrics: throughput, delivered data packet ratio, average end-to-end delay, jitter, overhead in the number of packets and overhead in the number of bytes. The simulation results show that protocol HACOR improves to AODV and OLSR in the throughput and delivered data packet ratio, showing intermediate values with respect to these two protocols in the metrics of average end-to-end delay, jitter, overhead in the number of packets and overhead in the number of bytes.

In PAntOR we have evaluated the following metrics: throughput, delivered data packet ratio, average end-to-end delay, jitter, overhead in the number of packets. The simulation results show that the protocol P-AntOR improves to AntOR-DNR in all the analyzed metrics.

In PAntOR-MI we have evaluated the following metric: delivered data packet ratio. The simulation results show that the protocol P-AntOR-MI improvement to its predecessor in very dynamic environments.

Chapter 8

Concluding Remarks and Future Work

This work has addressed a fundamental aspect of so-called mobile ad hoc networks as it is the routing problem.

First of all, we have seen the need to design specific routing protocols for mobile ad hoc networks because of the nature of them, as well as characteristics or requirements that they must meet to work properly, commenting also on the impossibility of using traditional solutions.

Secondly, it has analyzed a group of routing algorithms or protocols known as bioinspired which have their adaptive nature as a special feature, something particularly noteworthy in this type of environment. Within these algorithms, there has been particular reference in the literature of the concept of Swarm Intelligence, that is, those that apply the social behavior of insects and other animals to solve problems. The ACO algorithm is the starting point of these algorithms. The ACO algorithms are based on the collective behavior of ants in their search for food. ACO applies to a wide range of different problems. Due to its adaptability and robust properties, it also has become a paradigm for routing in mobile ad hoc networks. The ACO algorithms work iteratively. In every step artificial ants build a solution in parallel to the problem in question, using artificial pheromone matrix. Then the pheromone matrix on the basis of the solutions found is updated. In this way, pheromone matrix reflects information about good solutions that have been found until the date, and it allows the ants of later generations to utilize this information to create new ones.

Thirdly, we have performed a review of state of the art of the ACO routing protocols for mobile ad hoc networks observing that there are not representative protocols whose functioning metrics are degraded little or nothing in scalable environments. This review has included a comprehensive analysis of the AntHocNet protocol, the indisputable reference in the area. The study of literature has also picked up a compilation of the major parallelization techniques of ACO algorithms, which is especially interesting if you want to provide a scalable solution.

Subsequently, we have specified a new ACO routing protocol for mobile ad hoc networks called AntOR. Like its predecessor AntHocNet, AntOR is hybrid in the sense that it contains both reactive and proactive routing elements. In particular, it combines a reactive process of route setup with a proactive process of maintenance and exploration of new routes. Routing information is stored in pheromone tables that are similar to those utilized by other ACO routing algorithms. The forwarding of data and control packets is performed in a stochastic manner with the use of these tables. Link failures are treated with specific

reactive mechanisms, such as the local route repair and the use of warning messages. The key aspects of AntOR protocol are the use of disjoint node and disjoint link routes, the separation between the regular pheromone and virtual pheromone in the diffusion process and the exploration of new routes, which takes into account the number of hops in the best routes.

Then we have specified a family of ACO routing protocols for mobile ad hoc networks. All of them derived from the AntOR protocol, presenting two variants: the disjoint link version (AntOR-DLR) and the disjoint node version (AntOR-DNR). The disjoint Link version has originated a set of sequential protocols: AntOR-RDLR, AntOR-UDLR, AntOR-v2 and HACOR. All of these protocols are successive refinements from the original protocol. The disjoint node version has resulted in a set of parallel protocols: PAntOR and PAntOR-MI.

In AntOR-DNR the routes do not share nodes and in AntOR-DLR do not share links. By the disjoint property a failure in one node only affects a path, not the entire network. In addition, load balancing is better (by not to repeat routes). The routes calculation in AntOR-DLR is easier (less restrictive) than AntOR-DNR since all disjoint node is also a disjoint link, but not vice versa.

AntOR-RDLR differs from its predecessor (AntOR-DLR) in the pheromone update process and the route discovery mechanism,allowing the proactive forward ants to go by disjoint link routes until a maximum number of attempts has been made. This last allows the generation of more alternative routes.

The main idea of AntOR-UDLR is to replace link failure notification messages by unicast messages that are sent to the predecessor of the node that reports about the link failure until reaching the source of the data session, since in AntOR-DLR is sent in broadcast mode. The use of unicast messages makes losing fewer messages, because before transmitting it is checked if the medium is available through which you want to send, the fact that it does not happen when it is sent in broadcast mode. This new protocol aims to reduce network traffic, preventing the transmitted information gets unnecessarily nodes that do not need to process it.

AntOR-v2 and HACOR are the two more evolved variants, providing new optimization techniques such as storage of control packets and outdated routes management, as well as different failures link management and route exploration. The main difference between AntOR-v2 and HACOR is in the route exploration process and consists of the used technique. In AntOR-v2 proactive ants are sent to the 1-hop neighbor with best pheromone value. On the other hand, in HACOR the proactive process focuses on the algorithmic implementation S-ACO, which constitutes the starting point of the functioning of the ACO algorithms. HACOR also presents new techniques of link failure neutralization.

PAntOR is a large-grained parallelization version of AntOR making use of multiprocessor programming architectures based on a shared memory system through the standardization Posix Thread, which allows to execute tasks in parallel using threads, being applicable this parallelization in the route setup phase, local route repair process and link failure notification. PAntOR-MI is a multi-interface variant, which parallelizes the sending of broadcast messages by interface through threads.

Finally, various simulations have been performed in NS-3 in order to validate the earlier proposals. The simulation results show that: i) the AntOR-DLR protocol presented better throughput and better delivered data packet ratio than its predecessor AntHocNet as well as a small increase of average end-to- end delay and overhead, while these two latest metrics tend to equalize in dense networks; (ii) the AntOR-DLR protocol improves to AntOR-DNR; (iii) the AntOR-RDLR protocol improves to its predecessor AntOR-DLR;

(iv) the AntOR-UDLR protocol also improves to its predecessor AntOR-DLR in all metrics with the exception of the overload, in which case both protocols presented similar values; (v) the AntOR-v2 protocol improves to AODV in all the analyzed metrics, with the exception of the overload that is slightly superior, this difference becomes undetectable in dense networks; vi) HACOR protocol improves to AODV and OLSR in the throughput and the delivered data packet ratio, showing intermediate values for these two protocols in the other considered metrics; vii) the PAntOR protocol improves to AntOR-DNR; viii) the PAntOR-MI protocol improves to its predecessor in very dynamic environments; and ix) all specified protocols behave stably in all the simulations carried out.

The previous results allow us to conclude that the family of specified sequential protocols (of which HACOR is its greatest exponent) improves the scalability of its predecessor AntHocNet, protocol which is also better for most environments than the standards of reactive routing (AODV) and proactive (OLSR), and the considered parallel approaches for this type of ACO routing algorithms can further enhance the benefits of this family.

8.1 Future Work

Although the specified routing protocols provide an attractive solution for the design of a mobile ad hoc network, there are many issues which can arise. The main lines of research resulting from this work are:

- **Study of possible changes in AntOR**. It would be interesting to analyze other metrics in the route exploration process (end-to-end delay, end-to-end delay combined with the number of hops and so on); apply some pheromone evaporation process (AntOR does not have); use zone-disjoint routes [AEOP10] that, even if are more restrictive and independent than the disjoint link routes (or node), they tolerate best link failures; update disjoint routes in the route exploration and discovery phase in order to reduce the latency and overload and so on.

- **Design of new parallelization techniques**. It would be interesting to get more efficient parallel implementations. In PAntOR the sending is made to all neighbor using a thread for each one of them, which causes a high overload, even to collapse the devices. You could restrict this sending to a limited number of neighbors elected somehow (randomly, for example). Likewise we could use another parallelization technique distinct of shared memory through Posix Thread and to do some comparative. In PAntOR-MI is considered that paths which are created between pairs of origin/destination nodes take into account main address from each node, so that if a node receives the same control packet, it only takes into account the main IP address associated with that node. It could be considered to associate routes through intermediate nodes utilizing other interfaces that do not have associated the main address. With this possibility, a greater number of routes would be created in the route setup process.

- **Secure extension of the specified protocol**. All ACO routing protocols for mobile ad hoc networks are designed without considering the malicious behavior of some nodes, which can be exploited to violate the network security. AntOR belongs to this group of protocols where the attackers can alter its behavior; hence the need for an extension of this protocol, like it happens in the traditional routing protocols. In this sense it would be convenient to see the applicability of some of the most common security extensions developed for these as, for example, *Coded-Optimized*

Link State Routing (COD-OLSR) [GVGMRCSO11], secure extension of OLSR which adds a slight overload that barely affects the performance and that is an interesting alternative to provide integrity in OLSR against the classic mechanisms that make cryptography, more complex and with a large overload.

- **Application of the ACO routing protocols developed for mobile ad hoc networks in other fields**. An area of immediate application would be sensor networks [SDCF11] given the similarity between both types of networks. Another field of application could be robotic [DDCPG11] where frequently of the use of simple local interactions is analyzed to solve complex tasks such as, for example, navigation in indoor environments.

Bibliography

[Abr70] N. Abramson. The Aloha System: Another Alternative for Computer Communications. In *Proceedings of the Fall Joint Computer Conference*, volume 37, pages 281–285, Houston, Texas, USA, November 1970.

[AEOP10] K. Aburada, M. Eto, N. Okazaki, and M. Park. Proposal of a Zone Disjoint Multi-Path Routing for Ad Hoc Networks. In *Proceedings of the 8th Asia-Pacific Symposium on Information and Telecommunication Technologies*, pages 1–4, Kuching, Sarawak, Malaysia, June 2010.

[BHvR05] R. Barr, Z. J. Haas, and R. van Renesse. *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, chapter 19, pages 297–311. CRC Press, August 2005.

[BKS98] B. Bullnheimer, G. Kotsis, and C. Strauss. Parallelization Strategies for the Ant System. *Applied Optimization*, 24:87–100, June 1998.

[BM03] J. S. Baras and H. Mehta. A Probabilistic Emergent Routing Algorithm for Mobile Ad Hoc Networks. In *Proceedings of Modeling and Optimization in Mobile Ad Hoc Wireless Networks*, INRIA Sophia-Antipolis, France, March 2003.

[BR04] E. M. Belding-Royer. *Mobile Ad Hoc Networking*, chapter 10, pages 275–300. Wiley-IEEE Press, 1st Edition, June 2004.

[CE95] M. S. Corson and A. Ephremides. A Distributed Routing Algorithm for Mobile Wireless Networks. *Wireless Networks*, 1(1):61–81, February 1995.

[CG98] T. Chen and M. Gerla. Global State Routing: A New Routing Scheme for Ad Hoc Wireless Networks. In *Proceedings of the IEEE International Conference on Communications*, volume 1, pages 171–175, Atlanta, GA, USA, June 1998.

[CJ03] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, Internet Engineering Task Force, October 2003. http://www.ietf.org/rfc/rfc3626.txt.

[CP09] I. Chakeres and C. E. Perkins. Dynamic MANET On-Demand (DYMO) Routing. Internet Draft, Internet Engineering Task Force, March 2009. http://tools.ietf.org/html/draft-ietf-manet-dymo-17.

[DC04] G. A. Di Caro. *Ant Colony Optimization and Its Application to Adaptive Routing in Telecommunication Networks*. PhD thesis, Applied Sciences, Polytechnic School, Université Libre de Bruxelles, Brussels, Belgium, September 2004.

[DCD98a] G. A. Di Caro and M. Dorigo. AntNet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research*, 9(1):317–365, August 1998.

[DCD98b] G. A. Di Caro and M. Dorigo. Two Ant Colony Algorithms for Best-Effort Routing in Datagram Networks. In *Proceedings on Parallel and Distributed Computing and Systems*, pages 541–546, Las Vegas, Nevada, USA, October 1998.

[DCDG04] G. Di Caro, F. Ducatelle, and L. M. Gambardella. AntHocNet: An Ant-Based Hybrid Routing Algorithm for Mobile Ad Hoc Networks. In *Parallel Problem Solving from Nature*, volume 3242 of *Lecture Notes in Computer Science*, pages 461–470. Springer Berlin Heidelberg, September 2004.

[DCDG08] G. A. Di Caro, F. Ducatelle, and L. M. Gambardella. A Simulation Study of Routing Performance in Realistic Urban Scenarios for MANETs. In *Ant Colony Optimization and Swarm Intelligence*, volume 5217 of *Lecture Notes in Computer Science*, pages 211–218. Springer Berlin Heidelberg, September 2008.

[DDCG08] F. Ducatelle, G. A. Di Caro, and L. M. Gambardella. A New Approach for Integrating Proactive and Reactive Routing in MANETs. In *Proceedings of the 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, pages 377–383, Atlanta, GA, USA, October 2008.

[DDCG10] F. Ducatelle, G. A. Di Caro, and L. M. Gambardella. Principles and Applications of Swarm Intelligence for Telecommunications Networks. *Swarm Intelligence*, 4(3):173–198, September 2010.

[DDCPG11] F. Ducatelle, G. A. Di Caro, C. Pinciroli, and L. M. Gambardella. Self-organized Cooperation between Robotic Swarms. *Swarm Intelligence Journal*, 5(2):73–96, June 2011.

[DKGG01] P. Delisle, M. Krajecki, M. Gravel, and C. Gagné. Parallel Implementation of an Ant Colony Optimization Metaheuristic with OpenMP. In *Proceedings of the 3rd European Workshop on OpenMP*, Barcelona, Spain, September 2001.

[DMC96] M. Dorigo, V. Maniezzo, and A. Colorni. Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(1):29–41, February 1996.

[Dor92] M. Dorigo. *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano, Italy, Juanuary 1992.

[DS04] M. Dorigo and T. Stützle. *Ant Colony Optimization*. Bradford Company, 2004.

[Duc07] F. Ducatelle. *Adaptive Routing in Ad Hoc Wireless Multi-hop Networks*. PhD thesis, Università della Svizzera Italiana, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Lugano, Switzerland, May 2007.

[Fee01] L. M. Feeney. An Energy Consumption Model for Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks. *Mobile Networks and Applications*, 6(3):239–249, June 2001.

[FL01] J. A. Freebersyser and B. Leiner. *Ad Hoc Networking*, chapter 2, pages 29–51. Addison-Wesley, January 2001.

[GADP89] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. Self-Organized Shortcuts in the Argentine Ant. *Naturwissenschaften*, 76(12):579–581, December 1989.

[GHP02] M. Gerla, X. Hong, and G. Pei. Fisheye State Routing Protocol (FSR) for Ad Hoc Networks. Internet-draft, Internet Engineering Task Force, June 2002. http://tools.ietf.org/html/draft-ietf-manet-fsr-03.

[Gor00] D. Gordon. *Ants At Work: How An Insect Society Is Organized*. W. W. Norton & Company, September 2000.

[GSB02] M. Günes, U. Sorges, and I. Bouazizi. ARA - The Ant-Colony Based Routing Algorithm for MANETs. In *Proceedings of the 2002 International Conference on Parallel Processing Workshop on Ad Hoc Networks*, pages 79–85, Los Alamitos, California, USA, August 2002.

[GVGMRCSO11] L. J. García Villalba, J. García Matesanz, D. Rupérez Cañas, and A. L. Sandoval Orozco. Secure Extension to The Optimised Link State Routing Protocol. *IET Information Security*, 5(3):163–169, September 2011.

[GVRCSO10] L. J. García Villalba, D. Rupérez Cañas, and A.L. Sandoval Orozco. Bio-Inspired Routing Protocol for Mobile Ad Hoc Networks. *IET Communications*, 4(18):2187–2195, December 2010.

[GVRCSO13] L. J. García Villalba, D. Rupérez Cañas, and A. L. Sandoval Orozco. Parallel Approach of a Bioinspired Routing Protocol for MANETs. *International Journal of Ad Hoc and Ubiquitous Computing*, 12(3):141–146, March 2013.

[GVRCSOK12a] L. J. García Villalba, D. Rupérez Cañas, A. L. Sandoval Orozco, and T. Kim. Multiple Interface Parallel Approach of Bioinspired Routing Protocol for Mobile Ad Hoc Networks. *International Journal of Distributed Sensor Networks*, volume 2012, Article ID 532572:1–5, October 2012.

[GVRCSOK12b] L. J. García Villalba, D. Rupérez Cañas, A. L. Sandoval Orozco, and T. Kim. Restrictive Disjoint-Link-Based Bioinspired Routing Protocol for Mobile Ad Hoc Networks. *International Journal of Distributed Sensor Networks*, volume 2012, Article ID 956146:1–5, October 2012.

[HB03] M. Heissenbüttel and T. Braun. Ants-Based Routing in Large Scale Mobile Ad-Hoc Networks. In *Proceedings of Kommunikation in Verteilten Systemen*, pages 91–99, February 2003.

[HPS02] Z. J. Haas, M. R. Pearlman, and P. Samar. The Zone Routing Protocol (ZRP) for Ad Hoc Networks. Internet Draft, Internet Engineering Task Force, July 2002.
http://tools.ietf.org/id/draft-ietf-manet-zone-zrp-04.txt.

[HRFR06] T. R. Henderson, S. Roy, S. Floyd, and G. F. Riley. NS-3 Project Goals. In *Proceedings of the Workshop on NS-2: The IP Network Simulator*, Pisa, Italy, October 2006.

[HS03] O. Hossein and T. Saadawi. Ant Routing Algorithm for Mobile Ad Hoc Networks (ARAMA). In *Proceedings of the 22nd IEEE International Performance, Computing, and Communications Conference*, pages 281–290, Phoenix, Arizona, USA, April 2003.

[IEE99] IEEE802.11-1999. IEEE Standard for Local and Metropolitan Area Networks – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard Association, The Working Group for Wireless LAN, Edition of 1999. ratified in June 2003.

[Jai03] S. Jain. Energy Aware Communication in Ad-Hoc Networks. Technical Report UW-CSE 03-06-03, University of Washington, Computer Science and Engineering, Seattle, USA, January 2003.
ftp://ftp.cs.washington.edu/tr/2003/06/UW-CSE-03-06-03.pdf.

[JG07] G. Jayakumar and G. Gopinath. Ad Hoc Mobile Wireless Networks Routing Protocols – A Review. *Computer Science*, 3(8):574–582, 2007.

[JHM07] D. Johnson, Y. Hu, and D. Maltz. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. RFC 4728, Internet Engineering Task Force, February 2007.
http://www.ietf.org/rfc/rfc4728.

[KD08] B. Kalaavathi and K. Duraiswamy. Ant Colony Based Node Disjoint Hybrid Multi-Path Routing for Mobile Ad Hoc Network. *Journal of Computer Science*, 4(2):80–86, May 2008.

[KESMI+02] I. Kassabalidis, M. El-Sharkawi, R. J. Marks II, P. Arabshahi, and A. A Gray. Adaptive-SDR: Adaptive Swarm-Based Distributed Routing. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 2878–2883, Honolulu, Hawaii, USA, May 2002.

[KO08] S. Kamali and J. Opatrny. A Position Based Ant Colony Routing Algorithm for Mobile Ad-Hoc Networks. *Journal of Networks*, 3(4):31–41, April 2008.

[Kök08] M. M. Köksal. A Survey of Network Simulation Tools: Current Status and Future Developments, November 2008.
http://www.cse.wustl.edu/~jain/cse567-08/ftp/simtools/index.html.

[KV00] Y. Ko and N. H. Vaidya. Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. *Wireless Networks*, 6(4):307–321, July 2000.

[LF05] L. Liu and G. Feng. A Novel Ant Colony Based QoS-Aware Routing Algorithm for MANETs. In *Advances in Natural Computation*, volume 3612 of *Lecture Notes in Computer Science*, pages 457–466. Springer Berlin Heidelberg, August 2005.

[MAN] Mobile Ad Hoc Networks Work Group (MANET).
http://tools.ietf.org/wg/manet/.

[MC04] J. P. Macker and M. S. Corson. *Mobile Ad Hoc Networking*, chapter 9, pages 255–274. Wiley-Interscience, 1st Edition, October 2004.

[MGLA95] S. Murthy and J. J. Garcia-Luna-Aceves. A Routing Protocol for Packet Radio Networks. In *Proceedings of the 1st Annual ACM International Conference on Mobile Computing and Networking*, pages 86–95, Berkeley, CA, USA, November 1995.

[MM98] R. Michel and M. Middendorf. An Island Model Based Ant System with Lookahead for the Shortest Supersequence Problem. In *Parallel Problem Solving from Nature – PPSN V*, volume 1498 of *Lecture Note in Computer Science*, pages 692–701. Springer Berlin Heidelberg, September 1998.

[Moy98] J. Moy. OSPF Version 2. RFC 2328, Internet Engineering Task Force, April 1998.
http://tools.ietf.org/html/rfc2328.

[MTS02] S. Marwaha, C. K. Tham, and D. Srinivasan. Mobile Agents Based Routing Protocol for Mobile Ad Hoc Networks. In *Proceedings of the IEEE Global Telecommunications Conference*, volume 1, pages 163–167, Taipei, Taiwan, November 2002.

[NS2] The Network Simulator NS-2.
http://www.isi.edu/nsnam/ns.

[NS3] The Network Simulator NS-3.
http://www.nsnam.org.

[OMN] Objective Modular Network Testbed in C++ – OMNET++.
http://www.omnetpp.org/.

[OSP] Open Shortest Path First IGP Work Group (OSPF).
http://tools.ietf.org/wg/autoconf/.

[OTL04] R. Ogier, F. Templin, and M. Lewis. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). RFC 3684, Internet Engineering Task Force, February 2004.
http://www.ietf.org/rfc/rfc3684.

[OTT08] E. Osagie, P. Thulasiraman, and R. K. Thulasiram. PACONET: Improved Ant Colony Optimization Routing Algorithm for Mobile Ad Hoc Networks. In *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications*, pages 204–211, Okinawa, Japan, March 2008.

[PB94] C. E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of the ACM SIGCOMM Conference on Communications, Protocols and Applications*, pages 234–244, London, UK, September 1994.

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das. Ad Hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, Internet Engineering Task Force, July 2003. http://www.ietf.org/rfc/rfc3561.txt.

[PC01] V. Park and S. Corson. Temporally-Ordered Routing Algorithm (TORA), Version 1, Functional Specification. Internet Draft, Internet Engineering Task Force, July 2001. http://tools.ietf.org/html/draft-ietf-manet-tora-spec-04.

[PHM+06] R. Puttini, M. Hanashiro, F. Miziara, R. de Sousa, L. J. García Villalba, and C. J. Barenco Abbas. On the Anomaly Intrusion-Detection in Mobile Ad Hoc Network Environments. In *Proceedings of the 11th IFIP International Conference on Personal Wireless Communications*, pages 182–193, Albacete, Spain, September 2006.

[RAP10] M. K. Rafsanjani, S. Asadinia, and F. Pakzad. A Hybrid Routing Algorithm Based on Ant Colony and ZHLS Routing Protocol for MANET. In *Communication and Networking*, volume 120 of *Communications in Computer and Information Science*, pages 112–122. Springer Berlin Heidelberg, December 2010.

[RBR08] L. Rosati, M. Berioli, and G. Reali. On Ant Routing Algorithms in Ad Hoc Networks with Critical Connectivity. *Ad Hoc Networks*, 6(6):827–859, August 2008.

[RCGV12] D. Rupérez Cañas and L. J. García Villalba. Immune Systems for ACO-Based Routing Optimization. In *Proceedings of the 11th International Conference on Artificial Immune Systems*, Taormina, Italy, August 2012.

[RCGVSOK13] D. Rupérez Cañas, L. J. García Villalba, A. L. Sandoval Orozco, and T. Kim. Adaptive Routing Protocol for Mobile Ad Hoc Networks. *Computing*, pages 1–11, March 2013.

[RCSOGV11] D. Rupérez Cañas, A. L. Sandoval Orozco, and L. J. García Villalba. An Extension Proposal of AntOR for Parallel Computing. *Journal of Ubiquitous Systems & Pervasive Networks*, 3(2):67–72, October 2011.

[RCSOGV12a] D. Rupérez Cañas, A. L. Sandoval Orozco, and L. J. García Villalba. AntOR-UDLR: Aproximación Unicast de un Protocolo de Encaminamiento para Redes Móviles Ad Hoc. In *Actas del XXVII Simposium Nacional de la Unión Científica Internacional de Radio*, Elche, Alicante, Spain, September 2012.

[RCSOGV12b] D. Rupérez Cañas, A. L. Sandoval Orozco, and L. J. García Villalba. Technique to Neutralize Link Failures for an ACO-Based Routing Algorithm. In *Advances in Artificial Intelligence – IBERAMIA 2012*, volume 7637 of *Lecture Notes in Artificial Intelligence*, pages 251–260. Springer Berlin Heidelberg, November 2012.

[RCSOGV13a] D. Rupérez Cañas, A. L. Sandoval Orozco, and L. J. García Villalba. An Ant-Based Adaptive Distributed Routing Protocol for Mobile Ad Hoc Networks. In *Proceedings of the 6th International Conference on Information Technology*, Amman, Jordan, May 2013.

[RCSOGV13b] D. Rupérez Cañas, A. L. Sandoval Orozco, and L. J. García Villalba. Routing Techniques Based on Swarm Intelligence. In *Proceedings of the 7th International Conference on Intelligent Systems and Knowledge Engineering*, Advances in Intelligent Systems and Computing, Beijing, China, December 2013.

[RCSOGVH13] D. Rupérez Cañas, A. L. Sandoval Orozco, L. J. García Villalba, and P. Hong. HACOR: Hybrid ACO Routing Protocol for Mobile Ad Hoc Networks. *International Journal of Distributed Sensor Networks*, 2013:1–11, May 2013.

[RCSOGVK11a] D. Rupérez Cañas, A. L. Sandoval Orozco, L. J. García Villalba, and T. Kim. A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks. In *Multimedia, Computer Graphics and Broadcasting*, volume 263 of *Communications in Computer and Information Science*, pages 300–304. Springer Berlin Heidelberg, December 2011.

[RCSOGVK11b] D. Rupérez Cañas, A. L. Sandoval Orozco, L. J. García Villalba, and T. Kim. Comparing AntOR-Disjoint Node Routing Protocol with Its Parallel Extension. In *Multimedia, Computer Graphics and Broadcasting*, volume 263 of *Communications in Computer and Information Science*, pages 305–309. Springer Berlin Heidelberg, December 2011.

[RGLA99] J. Raju and J. J. Garcia-Luna-Aceves. A New Approach to On-Demand Loop-Free Multipath Routing. In *Proceedings of the 8th Annual IEEE International Conference on Computer Communications and Networks*, pages 522–527, Boston, MA, USA, October 1999.

[RL02] M. Randall and A. Lewis. A Parallel Implementation of Ant Colony Optimization. *Parallel and Distributed Computing*, 62(9):1421–1432, September 2002.

[RMH11] A. A. A. Radwan, T. M. Mahmoud, and E. H. Hussein. AntNet-RSLR: A Proposed Ant Routing Protocol for MANETs. In *Proceedings of Saudi International Electronics, Communications and Photonics Conference*, pages 1–6, Riyadh, Saudi Arabia, April 2011.

[RS06] S. Rajagopalan and C. Shen. ANSI: A Unicast Routing Protocol for Mobile Ad hoc Networks Using Swarm Intelligence. *Journal of Systems Architecture*, 52(8):485–504, August 2006.

[SDCF11] M. Saleem, G. A. Di Caro, and M. Farooq. A Review of Swarm Intelligence Based Routing Protocols for Wireless Sensor Networks. *Information Sciences*, 181(20):4597–4624, October 2011.

[SG07] J. Silberholz and B. L. Golden. The Generalized Traveling Salesman Problem: A New Genetic Algorithm Approach. In *Extending the Horizons: Advances in Computing, Optimization, and Decision Technologies*, volume 37 of *Operations Research/Computer Science Interfaces Series*, pages 165–181. Springer, 2007.

[Stü98] T. Stützle. Parallelization Strategies for Ant Colony Optimization. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, pages 722–731. Springer Berlin Heidelberg, September 1998.

[TG09] R. Thakur and W. Gropp. Test Suite for Evaluating Performance of Multithreaded MPI Communication. *Parallel Computing*, 35(12):608–617, December 2009.

[WDR08] M. Woo, N. H. Dung, and W. J. Roh. An Efficient Ant-Based Routing Algorithm for MANETs. In *Proceedings of the 10th International Conference on Advanced Communication Technology*, volume 2, pages 933–937, Gangwon-Do, Korea, February 2008.

[WOTT09] J. Wang, E. Osagie, P. Thulasiraman, and R. K. Thulasiram. HOPNET: A Hybrid Ant Colony Optimization Routing Algorithm for Mobile Ad Hoc Network. *Ad Hoc Networks*, 7(4):690–705, June 2009.

[WSJX07] Z. Wu, H. Song, S. Jiang, and X. Xu. Ant-Based Energy Aware Disjoint Multipath Routing Algorithm in MANETs. In *Proceedings of the International Conference on Multimedia and Ubiquitous Engineering*, pages 674–679, Seoul, Korea, April 2007.

[WvLW09] E. Weingärtner, H. vom Lehn, and K. Wehrle. A Performance Comparison of Recent Network Simulators. In *Proceedings of the IEEE International Conference on Communications*, Dresden, Germany, June 2009.

[ZGL04] X. Zheng, W. Guo, and R. Liu. An Ant-Based Distributed Routing Algorithm for Ad Hoc Networks. In *Proceedings of the International Conference on Communications, Circuits and Systems*, volume 1, pages 412–417, Chengdu, China, June 2004.

Part II

Resumen de la Investigación

En cumplimiento del artículo 4.3 de la normativa de desarrollo de los artículos 21 y 22 del R.D. 1393/2007 por el que se regulan los estudios universitarios oficiales de posgrado de la Universidad Complutense de Madrid, se presenta a continuación un resumen en español de la presente tesis que incluye introducción, objetivos, principales aportaciones y conclusiones del trabajo realizado.

Capítulo 9

Introducción

El término ad hoc es una locución latina que significa literalmente "para esto". Generalmente se refiere a una solución elaborada específicamente para un problema o fin preciso y, por tanto, no es generalizable ni utilizable para otros propósitos. Se usa pues para referirse a algo que es adecuado sólo para un determinado fin. En sentido amplio, ad hoc puede traducirse como "específico".

El Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) [IEE99] define las redes ad hoc como aquellas redes compuestas únicamente por estaciones, estando cada una de ellas dentro del rango de cobertura de alguna de las otras a través de un medio inalámbrico. Una red ad hoc se crea típicamente de manera dinámica y su principal singularidad es su limitación tanto temporal como espacial. Estas restricciones permiten crear y disolver redes de manera suficientemente sencilla y práctica. Formalmente, una red ad hoc inalámbrica presenta las siguientes características [Fee01]:

- **Inalámbrica**: Los nodos o estaciones se comunican a través de medios de transmisión no guiados (radio, infrarrojos, etc.).
- **Ad hoc**: La red es temporal y se establece dinámicamente de manera arbitraria por un conjunto de nodos según se necesita.
- **Autónoma y sin infraestructura**: La red no depende de ninguna infraestructura establecida ni de ninguna administración centralizada.
- **Multisalto**: No se necesitan encaminadores dedicados. Cada nodo actúa como encaminador y reenvía paquetes hacia otros nodos para facilitar el intercambio de información entre los integrantes de la red.

Adicionalmente, los nodos pueden estar dotados de movilidad. En este caso, estas redes reciben el nombre de redes móviles ad hoc (MANETs, Mobile Ad Hoc Networks). La topología de este tipo de redes es dinámica debido al constante movimiento de los nodos participantes, haciendo que los patrones de comunicación entre los miembros de la red evolucionen continuamente.

En definitiva, las redes móviles ad hoc eliminan las restricciones impuestas por las infraestructuras fijas, permitiendo a los dispositivos crear y adherirse a redes improvisadamente, haciéndolas adecuadas para adaptarse virtualmente a cualquier aplicación.

El resto de este capítulo está organizado como sigue: La sección 9.1 presenta el objeto de investigación de este trabajo. En la sección 9.2 se comenta brevemente la problemática del encaminamiento en redes móviles ad hoc. En la sección 9.3 se describe el contexto en el que se ha desarrollado la investigación realizada. En la sección 9.4 se presentan las

contribuciones de esta Tesis. Finalmente, la sección 9.5 resume la estructura del resto de la memoria.

9.1 Objetivos

Las redes móviles ad hoc presentan características especiales que deben tenerse en cuenta a la hora de implementar un protocolo de encaminamiento. Existen muchas soluciones (RFC 3626 [CJ03], RFC 3561 [PBRD03], 4728 [JHM07], 3684 [OTL04],...). Todos estos protocolos son soluciones válidas pero suelen tener como premisas de diseño unas determinadas características de topología y unos escenarios particulares, no siendo especialmente adecuados si hay cambios drásticos en la topología dinámica de la red ad hoc.

Existe un grupo de algoritmos o protocolos de encaminamiento denominados bioinspirados que tienen como característica esencial el hecho de ser adaptativos, algo especialmente reseñable en este tipo de ambientes. Dentro de estos algoritmos han sido especialmente referenciados en la literatura los basados en el concepto de inteligencia colectiva, esto es, aquellos que aplican el comportamiento social de los insectos y de otros animales para resolver problemas. El algoritmo Ant Colony Optimization (ACO) o algoritmo de optimización de la colonia de hormigas constituye el punto de partida de estos algoritmos. Los algoritmos ACO se basan en el comportamiento colectivo de las hormigas en su búsqueda del alimento y en llevarlo de vuelta al hormiguero.

AntHocNet [DC04, DCDG04, Duc07, WDR08, KO08, DCDG08] constituye un referente en el área de los protocolos de encaminamiento ACO para redes móviles ad hoc. Su carácter adaptativo le confiere unas propiedades especiales haciendo que sus métricas de funcionamiento, en términos generales, sean mejores que las de cualquier otro protocolo de encaminamiento para redes móviles ad hoc. No obstante lo anterior, en escenarios altamente dinámicos presenta problemas de escalabilidad. Este trabajo pretende subsanar tal deficiencia.

9.2 Identificación del Problema

El objetivo de un protocolo de encaminamiento para redes móviles es conseguir el envío de un mensaje de un nodo a otro sin existir un enlace directo. La mayoría de protocolos de encaminamiento para redes móviles ad hoc provienen de adaptaciones realizadas sobre protocolos de redes fijas, siendo su principal problema la cantidad de fallos que se producen en la comunicación debido a la movilidad de los nodos. Es necesario, por tanto, el diseño de algoritmos específicos que se adapten rápidamente a las peculiaridades de este tipo de redes.

9.3 Contexto de la Investigación

Esta tesis doctoral ha sido realizada dentro del grupo de investigación GASS (Grupo de Análisis, Seguridad y Sistemas, grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades de diversos proyectos de investigación que totalizan más de 5 años de trabajo.

Esta investigación se inicia en el contexto de un proyecto de investigación del Programa de Fomento de la Investigación Técnica (PROFIT) del Ministerio de Industria, Turismo y Comercio (MITyC) de la mano de la empresa Safelayer Secure Communications, S. A.,

empresa española que constituye un referente en el área de las Tecnologías de la Información y las Comunicaciones (área TIC) tanto en el ámbito nacional como internacional. Más concretamente, con el trabajo desarrollado en el proyecto Semantic & Ambient Trust Technologies (referencia FIT-360000-2007-48).

El Programa de Fomento de la Investigación Técnica es un instrumento mediante el cual el Gobierno articula un conjunto de convocatorias de ayudas públicas, destinadas a estimular a las empresas y a otras entidades a llevar a cabo actividades de investigación y desarrollo tecnológico; según los objetivos establecidos en el Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica (I+D+i), en la parte dedicada al Fomento de la Investigación Técnica.

La investigación prosigue en los proyectos: Semantic & Ambient Trust Technologies II (Subprograma Avanza I+D, referencia TSI-020100-2008-365), Semantic & Ambient Trust Technologies III (Subprograma Avanza I+D, referencia TSI-020100-2009-374), y Trust as a Service (Subprograma Avanza Competitividad I+D+I, referencia TSI-020100-2010-482), y finaliza con el proyecto Privacy-aware Accountability for a Trustworthy Future Internet (Subprograma Avanza Competitividad I+D+I, referencia TSI-020100-2011-165).

El Subprograma Avanza del MITyC, en sus diversas modalidades, viene a ser la continuación del Programa PROFIT. En todos los proyectos Avanza citados anteriormente también participa la empresa Safelayer Secure Communications, S. A.

9.4 Resumen de la Tesis

Desde el punto de vista del encaminamiento ACO en redes móviles ad hoc, los resultados de esta Tesis comprenden diversos protocolos (véase Figura 9.1).

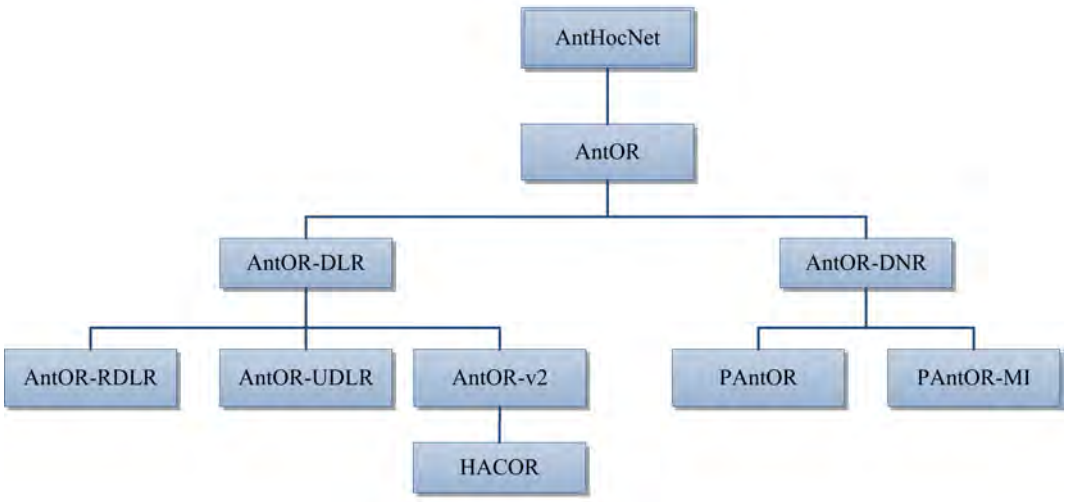


Figura 9.1: Esquema evolutivo de AntOR

Todos ellos derivan de un protocolo base denominado glsAntOR [GVRCSO10]. Como su predecesor AntHocNet, es híbrido en el sentido de que contiene elementos de encaminamiento tanto reactivos como proactivos, combinando un proceso reactivo de establecimiento de ruta con un proceso proactivo de mantenimiento y exploración de nuevas rutas. AntOR presenta dos variantes: AntOR disjunto de enlace o AntOR-DLR [RCSOGVK11b], donde las rutas no comparten enlaces, y AntOR disjunto de nodo o

ANTOR-DNR [RCSOGVK11b], donde las rutas no comparten nodos. Ambas variantes dan lugar a un conjunto de protocolos que son refinamientos sucesivos de las mismas.

La versión disjunta de enlace da lugar a un conjunto de protocolos secuenciales: AntOR-RDLR [GVRCSOK12b], AntOR-UDLR [RCSOGV12a, RCSOGV12b], AntOR-v2 [RCGV12, RCGVSOK13, RCSOGV13b] y HACOR [RCSOGV13a, RCSOGVH13]. AntOR-RDLR permite la generación de más rutas alternativas. AntOR-UDLR reduce el número de mensajes de control en los fallos de enlace. AntOR-v2 y HACOR son las variantes más evolucionadas, proporcionando nuevas técnicas como almacenamiento de paquetes de control y gestión de rutas obsoletas, así como mejoras en la gestión de fallos de enlace y en la exploración de rutas.

La versión disjunta de nodo da lugar a un conjunto de protocolos paralelos: PAntOR [RCSOGVK11b, GVRCSO13] y PAntOR-MI [GVRCSOK12a]. PAntOR es una versión paralelizada de AntOR que hace uso de arquitecturas multiprocesador de programación de grano grueso basadas en un sistema de memoria compartida por medio del estándar Posix Thread. PAntOR-MI es una variante multi-interfaz de PAntOR.

9.5 Estructura de la Tesis

Esta Tesis se estructura como sigue:

El Capítulo 10 realiza un estado del arte de las redes ad hoc incluyendo un repaso cronológico de la evolución de las redes ad hoc, un análisis de las características básicas de este tipo de redes, una presentación del protocolo de comunicación que actualmente se utiliza en ellas, una clasificación de las redes ad hoc y un resumen de las principales aplicaciones de las redes móviles ad hoc.

El Capítulo 11 aborda el problema del encaminamiento en redes móviles ad hoc. Comienza señalando la no aplicabilidad de las soluciones estándar y la necesidad de protocolos robustos y adaptativos. Posteriormente, presenta una clasificación de los protocolos de encaminamiento para redes móviles ad hoc, describiendo en detalle dos de singular importancia: OLSR y AODV.

El Capítulo 12 se centra en el algoritmo de optimización de la colonia de hormigas (ACO), área de la inteligencia artificial que se inspira en el comportamiento de las hormigas en la naturaleza y que es esencial para entender el funcionamiento de los protocolos de encaminamiento para redes móviles ad hoc desarrollados en la presente Tesis.

El Capítulo 13 analiza uno de las múltiples aplicaciones de la meta-heurística ACO: el denominado encaminamiento ACO que constituye todo un modelo en el diseño de protocolos de encaminamiento para redes móviles ad hoc. En este capítulo se revisa en detalle el estado del arte de los protocolos de encaminamiento ACO para redes móviles ad hoc, haciendo énfasis en AntHocNet, protocolo híbrido de encaminamiento ACO que es, sin duda alguna, un referente en el área.

El Capítulo 14 contiene las aportaciones de este trabajo: una familia de protocolos de encaminamiento ACO para redes móviles ad hoc construida a partir de un protocolo base denominado AntOR e inspirado en AntHocNet, del que hereda su carácter híbrido.

El Capítulo 15 contiene los resultados de las simulaciones realizadas en el software Network Simulator 3 (NS-3) [NS3].

Por último, el Capítulo 16 muestra las principales conclusiones extraídas de este trabajo así como algunas líneas futuras de investigación.

Capítulo 10

Redes Móviles Ad Hoc

El objetivo general de este capítulo es facilitar la comprensión de lo que son las redes móviles ad hoc. En primer lugar se hace un repaso cronológico de la evolución de las redes ad hoc. Posteriormente, se analizan las características básicas de este tipo de redes. Luego se presta atención al protocolo de comunicación que actualmente se utiliza en este tipo de redes, el IEEE 802.11. Seguidamente, se muestra una clasificación de las redes ad hoc. A continuación, se presentan las principales aplicaciones de las redes móviles ad hoc. El capítulo finaliza con una breve síntesis de lo expuesto en el mismo.

10.1 Evolución Histórica

En muy pocos años el campo de las redes ad hoc ha tenido una rápida expansión visible en la proliferación de dispositivos inalámbricos de bajo coste como ordenadores portátiles, asistentes personales digitales (PDAs), teléfonos móviles, etc.

A comienzos de los años 70 un trabajo pionero en radio de la Universidad de Hawai introduce el primer sistema que usa el medio de la radio para la transmisión de información. Conocido ampliamente como ALOHA [Abr70], fue desarrollado por Abramson y Kuo.

El trabajo realizado en Hawai llevó en 1972 al desarrollo de una arquitectura distribuida consistente en una red de difusión de radio con mínimo control central llamada PARNET bajo el patrocinio de DARPA. El proyecto ayudó a establecer el concepto de redes móviles ad hoc. PARNET permitía la comunicación directa entre usuarios móviles sobre grandes áreas geográficas, ancho de banda compartido y protección contra los efectos de múltiples caminos.

Los rápidos avances de la tecnología de la radio en los años 70 provocó la aparición de múltiples sistemas de comunicación móvil como teléfonos celulares e inalámbricos, sistemas de radio búsqueda, satélites móviles, etc.

Posteriormente, DARPA desarrolló el proyecto SURAN en 1983 que trata las tareas de escalabilidad de la red, seguridad, capacidad de proceso y gestión de energía. Se dedicaron esfuerzos para desarrollar dispositivos de bajo coste y con poco gasto de energía que pudieran soportar los avanzados protocolos de encaminamiento, escalar a miles de nodos las redes y dar soporte para ataques a la seguridad. El resultado fue la aparición de la tecnología conocida como LPR en 1987.

A mitad de los 90 se produce un nuevo avance con la llegada de las tarjetas de radio 802.11 para ordenadores personales y portátiles. En [FL01, Jai03] se propone por primera vez la idea de una colección de *hosts* móviles con una infraestructura mínima, y el IEEE acuña el término *redes ad hoc*.

Durante el mismo tiempo, el Departamento de Defensa de Estados Unidos continuaba trabajando con proyectos como el GloMo o el NTDR. El objetivo del GloMo era permitir la conectividad multimedia de tipo Ethernet, en cualquier momento y en cualquier lugar, entre los dispositivos inalámbricos. NTDR son protocolos que se basan en dos componentes: agrupamiento y encaminamiento. Los algoritmos de agrupamiento organizan dinámicamente una red en líderes y miembros de grupo. Los líderes forman la columna vertebral de la red y los miembros se comunican entre sí a través de dicha columna. NTDR inicialmente fue un prototipo para la Armada de los Estados Unidos y en la actualidad algunos países lo utilizan como base para otros protocolos.

La definición de estándares como IEEE 802.11 [IEE99] provocó el rápido crecimiento de las redes móviles en campos no sólo militares, sino también en el mundo comercial.

10.2 Características

Como su propio nombre indica la característica principal de una red móvil ad hoc es la movilidad de los nodos, que pueden cambiar de posición rápidamente. La necesidad de crear redes de forma rápida en lugares sin infraestructura suele implicar que los nodos exploren el área y, en algunos casos, se deban unir para conseguir un objetivo. El tipo de movilidad que desarrollen los nodos puede tener una influencia a la hora de elegir el protocolo de encaminamiento que aumente el rendimiento de la red.

Otro de los aspectos importantes en las redes ad hoc es la llamada auto-organización que se estudia en profundidad en [Fee01]. La idea principal se basa en la coordinación y colaboración de todos los nodos de la red para conseguir un mismo objetivo. Se han propuesto varios métodos de auto-organización para redes en general y para redes ad hoc en particular.

La auto-organización puede desglosarse en las capacidades mostradas en la Tabla 10.1.

Tabla 10.1: Capacidades de la auto-organización

Capacidad	**Descripción**
Auto-reparación	Mecanismos que permitan detectar, localizar y reparar automáticamente los fallos siendo capaces de distinguir la causa del error. Por ejemplo, sobrecarga o mal funcionamiento.
Auto-configuración	Métodos de generación de configuraciones adecuadas en función de la situación actual dependiendo de las circunstancias ambientales. Por ejemplo, conectividad o parámetros de calidad de servicio.
Auto-gestión	Capacidad de mantener dispositivos o redes dependiendo de los parámetros actuales del sistema.
Adaptación	Adecuación a los cambios de las condiciones ambientales. Por ejemplo, cambio en el número de nodos vecinos.

A continuación se presentan el resto de características de las redes móviles ad hoc:

- **Ausencia de infraestructura**: Al contrario que las redes convencionales que cuentan con la existencia de elementos físicos, las redes móviles se forman autónomamente.
- **Topología dinámica**: Los nodos se pueden mover arbitrariamente haciendo que algunos enlaces se destruyan y otros se creen cuando un nodo se acerque a otros que antes tenía fuera de su alcance.
- **Ancho de banda limitado**: En la mayoría de las ocasiones será menor que el de una conexión cableada, afectado además por las interferencias de las señales electromagnéticas.
- **Variación en la capacidad de los enlaces y los nodos**: Los nodos pueden disponer de varias interfaces de radio que difieren entre sí en capacidad de transmisión/recepción y en la banda de frecuencia en la que trabajan. Esta característica complica el desarrollo de los protocolos de encaminamiento en gran medida.
- **Conservación de energía**: Algunos o todos los nodos de una red móvil ad hoc son alimentados por baterías y no tienen posibilidad de recargarlas. Para estos nodos el criterio más importante a la hora de diseñar sistemas y protocolos será la optimización de la conservación de energía.
- **Escalabilidad**: En muchas aplicaciones las redes ad hoc pueden llegar a tener miles de nodos lo que conlleva dificultad en tareas como direccionamiento, encaminamiento, gestión de localización, gestión de configuración, interoperabilidad, seguridad, etc.
- **Falta de seguridad**: La seguridad juega un papel importante en las redes ad hoc dado el carácter vulnerable de los enlaces inalámbricos que se forman. Los protocolos de encaminamiento deben proporcionar una comunicación segura. Existen áreas de investigación en este sentido que sugieren incluir datos de sensores externos e información geográfica y topográfica en el propio algoritmo de encaminamiento.
- **Encaminamiento multisalto**: Los nodos actúan como encaminadores para retransmitir los paquetes intercambiados entre nodos cuyo alcance no permite una comunicación directa.
- **Entorno imprevisible**: Las redes ad hoc pueden darse en terrenos en los que las situaciones no son las más óptimas debido a condiciones peligrosas o desconocidas. Pueden darse casos donde los nodos se destruyan, se estropeen o comiencen a producir fallos.
- **Comportamiento de los terminales**: Una de las principales claves para que una red móvil ad hoc tenga un funcionamiento adecuado es la confianza que cada nodo debe tener sobre los demás. Sin esta confianza sería imposible crear un protocolo de encaminamiento ya que la información debe transmitirse por varios nodos intermedios. Normalmente, los protocolos de encaminamiento que descubren los terminales intermedios se basan en las respuestas que dan los nodos sobre el coste de la comunicación. Existen nodos maliciosos que podrían intencionadamente informar de forma incorrecta sobre los costes con la finalidad de recibir todos los paquetes, poder manipularlos, alterarlos o incluso eliminarlos. Algunas soluciones al respecto se encuentran en [PHM+06].

La Figura 10.1 presenta un ejemplo típico de una red móvil ad hoc.

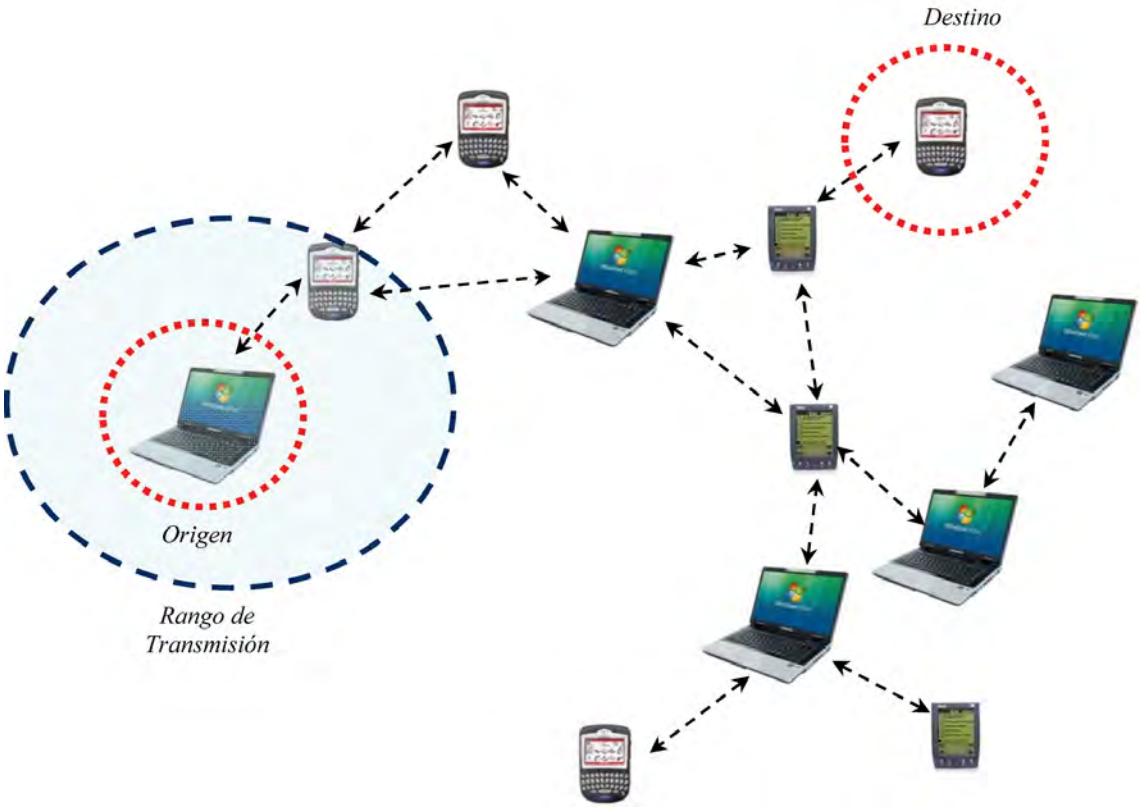


Figura 10.1: Red móvil ad hoc

10.3 Estándar IEEE 802.11

El IEEE 802.11 es un estándar de protocolo de comunicaciones que define el uso de los dos niveles más bajos de la arquitectura *Open System Interconnection* (OSI), capa física y capa de enlace de datos, especificando sus normas de funcionamiento en una red inalámbrica. La primera propuesta de este estándar mantenía tasas de transmisión de 1 y 2 Mbps en la banda de frecuencias *Industrial Scientific and Medical* (ISM), situada en 2.4 GHz. Además, se especificaban como tecnologías en la capa física los infrarrojos y el canal radio. Con los años se ha llegado a distintas versiones del estándar. Se citan los más importantes a continuación:

- **IEEE 802.11a**: Hasta 54 Mbps a 5 GHz. Utiliza la tecnología *Orthogonal Frecuency-Division Multiplexing* (OFDM) en la capa física.
- **IEEE 802.11b**: Hasta 11 Mbps a 2.4 GHz. Actualmente es el más utilizado. Utiliza la tecnología *Direct Sequence Spread Spectrum* (DSSS) en la capa física.
- **IEEE 802.11e**: Pretende proporcionar *Quality of Service* (QoS) para su uso en servicios como *Voice-over Internet Protocol* (VoIP) y *Streaming*. Una aproximación para otorgar calidad de servicio es la de diferenciar los paquetes clasificándolos en un número pequeño de tipos de servicios y utilizar mecanismos de prioridad para proporcionar una calidad de servicio adecuada a cada tráfico.
- **IEEE 802.11f**: Desarrolla especificaciones para la implementación de puntos de

acceso y sistemas de distribución para evitar problemas de interoperabilidad entre distintos fabricantes y distribuidores de equipos.

- **IEEE 802.11g**: Hasta 54 Mbps a 2.4 GHz. Soporta tanto OFDM como DSSS en la capa física.

10.4 Clasificación

La terminología de redes ad hoc aún no está muy asentada y no existe una clasificación clara. A continuación se exponen varias clasificaciones situando el lugar en el que se encuentran las redes móviles ad hoc.

Existen redes ad hoc *con infraestructura* donde los nodos se mueven mientras se comunican con una estación base fija. Cuando un nodo se mueve fuera del rango de una estación fija entra en el alcance de otra estación. Por otro lado, se encuentran las redes ad hoc *sin infraestructura* donde no existen estaciones base fijas y todos los nodos de la red necesitan actuar como *routers*. **Las redes móviles ad hoc son redes ad hoc sin infraestructura**.

Otra clasificación de las redes ad hoc incluye las *redes de un solo salto* y las *redes multisalto*. Los nodos de las redes de un solo salto se comunican únicamente con los nodos que tienen a su alcance. En las redes ad hoc multisalto los nodos que no pueden comunicarse directamente utilizan nodos intermedios para retransmitir la información. **Las redes móviles ad hoc son redes ad hoc multisalto**.

Por último hay una clasificación que incluye las redes móviles ad hoc como un tipo independiente. Se incluyen tres tipos de redes ad hoc:

- **Redes móviles ad hoc**.
- **Redes de sensores**: También denominadas *Wireless Sensor Networks* (WSN). Formadas de dispositivos sensoriales, generalmente compuestos por un sensor tradicional y un conversor analógico-digital. La unidad de proceso está compuesta de un microprocesador y una pequeña memoria. Pueden incluir sistemas de localización y sistemas de movilidad. En estas redes el número de nodos suele ser mucho mayor que en una red móvil ad hoc pero la movilidad se considera escasa o nula (solamente cambia la topología con la pérdida o desconexión de nodos). Es habitual el flujo de información desde muchos orígenes hasta un nodo llamado sumidero (*sink*) que se encarga de procesar la información y enviarla al destino.
- **Redes híbridas**: También denominadas mixtas, son redes ad hoc que usan infraestructuras IP si están disponibles.

A su vez, las redes móviles ad hoc se pueden dividir en dos tipos en función de si están conectadas o no a otras redes:

- **Redes móviles ad hoc autónomas**: Son redes que no están conectadas a ninguna otra red. Los nodos de la red se pueden identificar unívocamente a través de una dirección IP con la única premisa de que sea distinta a la de cualquier otro nodo de la red.
- **Redes móviles ad hoc subordinadas**: Son redes conectadas a una o más redes externas. Se obliga a usar un direccionamiento IP topológico correcto y encaminable globalmente. Un ejemplo típico de red móvil ad hoc subordinada es una red móvil ad hoc que es parte de Internet.

10.5 Aplicaciones

Es fácil encontrar situaciones donde se ve la utilidad de las redes móviles ad hoc. Uno de los ejemplos más clásicos (aunque también discutido) es una reunión de trabajo: un grupo de personas con ordenadores portátiles o PDA. Son de distintas empresas y por tanto sus direcciones son distintas. Tal vez en la sala haya acceso a Internet y puedan usar por ejemplo IP móvil, pero ¿para qué pasear sus datagramas por toda la ciudad o todo el país cuando están en la misma habitación? Sus equipos probablemente estén dotados de puertos de infrarrojos o Bluetooth que les permitan formar una red para la ocasión. En algunos casos, simplemente no habrá infraestructuras de apoyo. Por ejemplo, en poblaciones aisladas o de orografía difícil, situaciones de emergencia, desastres naturales donde las infraestructuras hayan desaparecido, etc.

Otro ejemplo son las denominadas *Personal Area Network* (PAN), redes formadas por los dispositivos de una persona, como su reloj, su agenda y su teléfono móvil. Una red así puede querer entrar en contacto con la red de otra persona que en ese momento esté próxima.

La capacidad de desplegarse inmediatamente y la no dependencia de un único punto de fallo hace a estas redes muy interesantes para el uso militar. El campo militar es posiblemente el más desarrollado actualmente. Así, el ejército estadounidense ya dispone de un sistema basado en este tipo de redes, el *Force XXI Battle Command, Brigade-and-Below* (FBCB2). Uno de sus objetivos es distinguir las fuerzas propias de las fuerzas del enemigo, ofreciendo a los soldados una visión del campo de batalla similar a la de un videojuego. Los equipos de la generación inmediatamente anterior estaban basados en comunicaciones por satélite, con latencias de cinco minutos. En abril de 2003 el FBCB2 se utilizó en la Segunda Guerra del Golfo, lo que supuso probablemente el primer uso bajo fuego real de una red móvil ad hoc.

Otro motivo por el que una red móvil ad hoc puede ser ventajosa es el coste. Aunque exista una infraestructura de red, si pertenece a una entidad ajena es muy posible que cobre por su uso, mientras que si están los equipos desplegados se dispondrá ya de una red sin coste adicional. Por ejemplo, los coches que pasan por una autopista podrían formar fácilmente una red móvil ad hoc, independientemente de su capacidad de conectarse a otras redes como *Global System for Mobile Comunications* (GSM) o similar.

Por último, supóngase que se tienen estaciones capaces de comunicarse empleando un satélite. Estos equipos de comunicaciones son caros, pero bastaría con que algunos tengan capacidad de conectarse al satélite para que todos dispusieran de conectividad. Y no todos los capaces de conectarse al satélite necesitarían estar conectados simultáneamente.

La cualidad más notable de las redes ad hoc es su flexibilidad. El hecho de que puedan establecerse en cualquier lugar y en cualquier momento sin infraestructura, administración o preconfiguración, las hace muy atractivas para un amplio rango de campos de aplicación.

La Tabla 10.2 muestra una clasificación de las aplicaciones presentes y futuras de las redes ad hoc, así como de los servicios que ofrecen [BR04].

Tabla 10.2: Aplicaciones de las redes móviles ad hoc

Redes tácticas	Comunicaciones en operaciones militares.
	Campos de batalla automatizados.
Redes de sensores	Recogida de datos en tiempo real, generalmente altamente correlados en espacio y tiempo.
Servicios de salvamento	Operaciones de búsqueda y rescate.
y emergencia	Sustitución de redes con infraestructuras en situaciones de catástrofes naturales.
Entornos comerciales	Comercio electrónico.
	Acceso remoto a los registros de los clientes desde una base de datos centralizada.
	Oficina móvil.
	Servicios vehiculares.
Redes para particulares	WLAN en hogares y oficinas.
y redes para empresas	Redes de área personal (PAN).
Aplicaciones educativas	Configuración de comunicaciones ad hoc en reuniones, conferencias y congresos.
	Configuración de clases virtuales.
Ocio	Juegos multi-usuario.
	Robots mascota.
	Acceso a Internet en exteriores.
Servicios de localización	Servicios de seguimiento.
	Servicios de información.

10.6 Resumen

El objetivo de este capítulo ha sido introducir las redes móviles ad hoc. Se ha comenzado con un breve repaso de la evolución de las mismas. Posteriormente, se han analizado sus principales características entre las que destaca la ausencia de infraestructura, la topología dinámica y la capacidad de auto-organización. También se han comentado las distintas versiones del estándar IEEE 802.11 o protocolo de comunicación de este tipo de redes. Finalmente, se ha visto una clasificación de las redes móviles ad hoc, así como que su flexibilidad las hace idóneas para un gran número de aplicaciones.

Capítulo 11

Encaminamiento en Redes Móviles Ad Hoc

Este capítulo está dedicado a la problemática del encaminamiento en redes móviles ad hoc, un aspecto fundamental de este tipo de redes. Se comienza viendo la necesidad del diseño de protocolos de encaminamiento específicos para las redes móviles ad hoc dada la naturaleza de las mismas, así como las características o requisitos que deben cumplir para funcionar adecuadamente, comentándose también la imposibilidad de utilizar las soluciones tradicionales. Posteriormente, se muestran diversas clasificaciones de los protocolos de encaminamiento para redes móviles ad hoc: en función de la información de estado que almacenan los nodos de la red, en función de la estructura y en función del procedimiento adoptado para el descubrimiento del camino a establecer. A continuación, se presta especial atención a los protocolos AODV y OLSR, una referencia en el área, que se van a utilizar en el análisis de las prestaciones de los protocolos de encaminamiento desarrollados en el presente trabajo. El capítulo finaliza con una breve síntesis de lo expuesto en el mismo.

11.1 Protocolos de Encaminamiento

En redes móviles ad hoc los protocolos convencionales o bien tendrán un rendimiento muy pobre, o bien serán simplemente inaplicables. Como alternativa se desarrollan protocolos específicos de encaminamiento. Con frecuencia se les denomina de nivel 2.5, ya que es habitual encontrarlos por encima de protocolos de enlace como IEEE 802.11 y por debajo del protocolo de red IP.

El concepto de encaminamiento básicamente comprende dos actividades. En primer lugar, determinar los caminos óptimos y, en segundo lugar, transferir los grupos de paquetes de información a través de la red. Los algoritmos utilizan varias métricas para calcular el mejor camino para que los paquetes lleguen a su destino. Estas métricas son medidas estándar como podría ser el número de saltos que son usados por el algoritmo para determinar el camino óptimo. El proceso para determinar el camino inicializa y mantiene tablas de encaminamiento que contienen la información total de cada ruta. La información que se almacena para cada ruta varía de un algoritmo a otro.

Las redes móviles ad hoc se construyen de forma dinámica cuando un conjunto de nodos crean rutas entre sí para conseguir la conectividad entre ellos. Los nodos de la red móvil ad hoc pueden actuar como origen o destino de una comunicación, pero también como encaminadores cuando una relación entre nodos no se puede realizar directamente por motivos de alcance. De esta forma se crean comunicaciones multisalto. Un protocolo de encaminamiento de una red móvil ad hoc necesita proveer un mecanismo que mantenga

las rutas hacia los destinos frente al movimiento de los nodos que puede provocar que las rutas se destruyan, y sea necesario encontrar una ruta alternativa para mantener la comunicación entre los nodos.

El objetivo de un protocolo de encaminamiento para redes móviles es conseguir el envío de un mensaje de un nodo a otro sin existir un enlace directo. La mayoría de protocolos de encaminamiento para redes móviles ad hoc provienen de adaptaciones realizadas sobre protocolos de redes fijas, siendo su principal problema la cantidad de fallos que se producen en la comunicación debido a la movilidad de los nodos.

Los protocolos de encaminamiento para redes móviles ad hoc deben satisfacer básicamente los siguientes criterios [BR04]:

- *Señalización mínima*: La reducción de los mensajes de control ayuda a conservar la capacidad de las baterías y la comunicación de los nodos.
- *Mantenimiento dinámico de topología*: El algoritmo deberá ser capaz de localizar una nueva ruta rápidamente cuando se rompe un enlace.
- *Libre de bucles*: Se pretende evitar el problema de tener paquetes circulando perdidos por la red.
- *Capacidad multisalto*: Debe asegurarse el reenvío de paquetes a través de los nodos de la red dado que habitualmente el destino no se encuentra dentro del alcance de la fuente.
- *Tiempo de procesamiento mínimo*: Se requieren algoritmos con cálculos computacionales que no sean excesivamente complejos para disminuir el tiempo de procesamiento y alargar de esta forma el tiempo de vida de la batería.

Además, debe admitir diversos modos de operación [MC04]:

- *Distribuido*: Propiedad esencial de las redes MANET.
- *Inactivo*: Los protocolos de encaminamiento deberán estar preparados para afrontar aquellos períodos de tiempo en los cuales los nodos frenan su actividad y permanecen inactivos para ahorrar energía.
- *Bajo demanda*: La adaptación del encaminamiento a los patrones de tráfico particulares de cada situación hace posible reducir el gasto de ancho de banda y energía, aunque se amplía el tiempo de obtención de la ruta.
- *Soporte de enlaces unidireccionales*: Los protocolos de encaminamiento en muchas ocasiones han sido diseñados y funcionan correctamente sólo con enlaces bidireccionales y esto no debería ser así, porque en la práctica podemos encontrarnos con la existencia de enlaces unidireccionales que sean clave para el intercambio de información en redes móviles ad hoc.

Se han diseñado numerosos protocolos de encaminamiento para redes MANET atendiendo a estos criterios.

La finalidad del MANET WG [MAN] del IETF es estandarizar la funcionalidad de un protocolo de encaminamiento IP para aplicaciones de encaminamiento inalámbrico dentro de topologías tanto estáticas como dinámicas como consecuencia de la movilidad de los nodos u otros factores. Los enfoques están destinados a ser relativamente generales pues deben ser adecuados en múltiples entornos inalámbricos y hardware y dirigidos a escenarios

donde las redes móviles ad hoc estén desplegadas en la frontera de una infraestructura IP. Las infraestructuras *mesh* híbridas (por ejemplo, una mezcla de *routers* móviles y fijos) deberían ser soportados por las especificaciones de las redes móviles ad hoc.

11.2 Clasificación de los Protocolos de Encaminamiento

Desde que se empezaron a estudiar las redes móviles ad hoc se han propuesto diversas clasificaciones de los protocolos de encaminamiento que se resumen en [JG07].

En función de la información de estado que almacenan los nodos de la red los protocolos pueden ser clasificados en protocolos basados en la topología y protocolos basados en el destino. En los primeros cada nodo toma decisiones basándose en una completa información de la topología de la red. Los segundos son protocolos que manejan vectores de distancias, en los que cada nodo intercambia con sus vecinos las distancias que conoce a otros nodos.

Otra clasificación propone dividir los protocolos en función de la estructura, diferenciándose varios niveles. La Figura 11.1 presenta esta clasificación formulada, especificando algunos de los protocolos representativos de cada categoría:

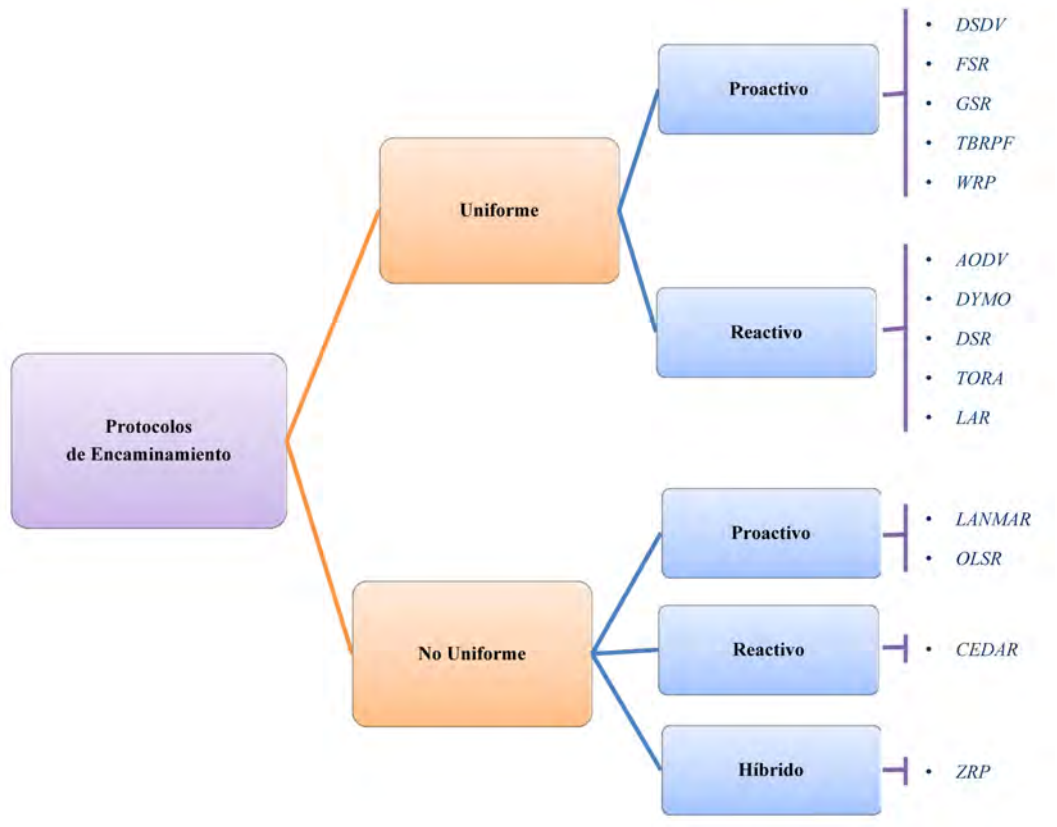


Figura 11.1: Taxonomía de protocolos de encaminamiento en redes móviles ad hoc

El primer nivel se refiere a la homogeneidad o heterogeneidad de las funciones de los nodos en el encaminamiento, distinguiéndose dos tipos:

- *Protocolos uniformes o de estructura plana*: Ningún nodo de la red realiza un papel

distinto al de los demás, todos ellos envían y responden a los mensajes de control del mismo modo.

- *Protocolos no uniformes*: Típicos de estructuras jerárquicas en las que algunos nodos desarrollan papeles especiales e incluso pueden dotarse de capacidades particulares en términos de cómputo, energía o almacenamiento entre otros. Esto les permite soportar algoritmos más complejos, reducir la sobrecarga debida a la comunicación y ofrecer la posibilidad de balanceo de carga mientras mantienen sus características, incluso ante incrementos del número de nodos en la red. Por el contrario, generan cierto coste de mantenimiento de la estructura y necesitan en muchos casos la disponibilidad de nodos heterogéneos.

Dentro de estas dos categorías anteriores, los protocolos presentan una nueva peculiaridad relativa al procedimiento adoptado para el descubrimiento del camino a establecer y su mantenimiento. Esta clasificación, sin duda, es la más difundida surgiendo los siguientes tipos de protocolos:

- *Protocolos proactivos*: En este tipo de encaminamiento cada nodo mantiene información de cómo llegar a cualquier otro nodo de la red e intercambia esta información con todos sus vecinos. La información de encaminamiento es normalmente almacenada en un número diferente de tablas. Periódicamente se actualizan las tablas si la topología de red cambia. La diferencia entre los protocolos de este tipo se encuentra en la forma de actualizar y detectar la información de encaminamiento y el tipo de información que se guarda en cada tabla. La ventaja que aportan estos protocolos es la baja latencia ya que las rutas están siempre disponibles. Sin embargo, esto conlleva un consumo de energía muy alto en los nodos y se puede producir una sobrecarga de mensajes en la red debido a la inundación periódica de mensajes. Seguidamente se enumeran los protocolos de encaminamiento proactivos más representativos: *Destination-Sequenced Distance-Vector* (DSDV) [PB94], *Wireless Routing Protocol* (WRP) [MGLA95], *Global State Routing* (GSR) [CG98], *Fisheye State Routing Protocol* (FSR) [GHP02], *Optimized Link State Routing* (OLSR) [CJ03], *Topology Broadcast Reverse Path Forwarding*(TBRPF) [OTL04]. En general, estos protocolos tratan de evitar bucles en las rutas, consumo excesivo de memoria y reducción del tamaño de los paquetes que contienen la información de las tablas de encaminamiento. Dentro de los protocolos proactivos se pueden distinguir dos subtipos de protocolos según su comportamiento: los conducidos por eventos (*event-driven*), que envían paquetes con información sobre las rutas sólo cuando éstas sufren algún cambio, y los que refrescan la información periódicamente (*regular updated*). El protocolo OLSR, que ha sido utilizado para este trabajo y que será analizado en detalle en el siguiente apartado, entra dentro de la segunda categoría.

- *Protocolos reactivos*: Estos protocolos tratan de reducir la sobrecarga que producen los protocolos proactivos. Para ello proponen que los nodos de la red móvil ad hoc, cuando no tienen una ruta a un destino, la calculen sólo cuando es necesaria, es decir, cuando el nodo tenga que comenzar un intercambio de paquetes con el destino. El descubrimiento de una ruta normalmente se realiza por inundación de mensajes de solicitud por toda la red. Estos protocolos conllevan una alta latencia, provocada por el descubrimiento de rutas. Sin embargo, la sobrecarga de mensajes por la red se reduce. Seguidamente se enumeran los protocolos de encaminamiento reactivos más representativos: *Ad hoc On-demand Distance Vector* (AODV) [PBRD03], *DYnamic Manet On-demand* (DYMO) [CP09], *Dynamic Source Routing* (DSR) [JHM07], *Routing On-demand Acyclic Multi-path* (ROAM) [RGLA99], *Lightweight Mobile Routing*

(LMR) [CE95], *Location-Aided Routing* (LAR) [KV00], *Temporally-Ordered Routing Algorithm* (TORA) [PC01]. La mayoría de ellos tienen el mismo coste de encaminamiento en el peor escenario posible ya que casi todos siguen la misma filosofía para el descubrimiento de rutas.

- *Protocolos híbridos*: Combinando los protocolos proactivos y reactivos nacen los protocolos híbridos que pretenden minimizar los inconvenientes de ambos. La idea de estos protocolos es que los nodos de la red trabajen de forma proactiva con los nodos más cercanos y de forma reactiva con el resto de nodos. La parte reactiva controla la sobrecarga y el consumo de memoria al calcular las rutas sólo cuando son necesarias. En contraste, la parte proactiva necesita actualizar periódicamente la información almacenada y mantiene rutas que quizás nunca serán utilizadas, añadiendo una innecesaria sobrecarga. El caso más conocido de protocolo híbrido es *Zone Routing Protocol* (ZRP) [HPS02].

El Grupo de Trabajo MANET tiene previsto desarrollar dos especificaciones de protocolo de encaminamiento estándar, denominadas *Reactive MANET Protocol* (RMP) y *Proactive MANET Protocol* (PMP), si bien también puede decidir un enfoque mixto. Soportará IPv4 e IPv6, requisitos de seguridad y otros aspectos, y prestará atención especial al protocolo OSPF-MANET que viene desarrollando el OSPF WG [OSP]. El OSFP WG desarrolla extensiones del protocolo OSPF para diferentes escenarios, siendo OSPF-MANET la extensión de OSPF a redes móviles ad hoc.

En los siguientes apartados se describe un ejemplo representativo de protocolo proactivo, reactivo e híbrido, más concretamente se explica el funcionamiento del protocolo proactivo OLSR, del protocolo reactivo AODV y del protocolo híbrido ZRP. Se hace énfasis en los dos primeros ya que en la literatura se utilizan como referencia para comparar las prestaciones de los nuevos protocolos de encaminamiento.

11.3 Protocolo OLSR

El protocolo de encaminamiento *Optimized Link State Routing* (OLSR) pertenece al grupo de los protocolos para MANETs que están definidos como RFC (Request For Comments). OLSR se especifica en la RFC 3626 [CJ03], siendo una optimización del clásico protocolo de estado de enlace u *Open Shortest Path First* (OSPF) [Moy98], pero adaptado a redes móviles ad hoc.

Al tratarse de un protocolo proactivo la información sobre las rutas hacia todos los nodos se mantiene siempre actualizada, para que esté disponible en caso de que sea necesaria. Como se ha indicado anteriormente, OLSR es *regular updated*, esto es, cada cierto tiempo paquetes de información sobre las rutas son transmitidos, aunque no se hayan detectado cambios.

La principal aportación de OLSR que lo diferencia de otros protocolos similares son las optimizaciones que se realizan para que la sobrecarga producida por las actualizaciones periódicas sea mínima. El modo en que se hace llegar la información sobre rutas a toda la red es mediante inundación controlada: designando a ciertos nodos encargados de enviarse entre ellos la información, haciéndola llegar posteriormente al resto de la red, y comprobando que no se envían datos duplicados.

Debido a las optimizaciones que aplica, obtiene buenos resultados en redes grandes y densas. Sus optimizaciones se notan en el rendimiento cuanto más grandes sean las redes, sobre todo si el tráfico es esporádico entre pares de nodos que varíen irregularmente, en lugar de comunicaciones regulares entre nodos concretos.

11.3.1 Funcionamiento del Protocolo

Lo primero que hace un nodo al iniciarse es detectar con qué otros nodos tiene conexión a nivel de enlace. Para ello periódicamente se emiten mensajes *Hello*. Estos mensajes no se retransmiten por los nodos que los reciben, ya que su finalidad es que los nodos se den a conocer a sus vecinos de un salto, es decir, nodos con los que existe conectividad a nivel de enlace. Otra funcionalidad de los mensajes *Hello*, aparte de dar a conocer al propio nodo, es anunciar los vecinos ya conocidos del nodo emisor. De esta forma, un nodo que escucha estos mensajes no sólo descubre a sus vecinos de un salto, sino que además adquiere conocimiento de sus vecinos de dos saltos de distancia. La clave del protocolo está en los Multipuntos de Retransmisión (*Multipoint Relay*), abreviadamente MPR. Una vez que un nodo conoce el conjunto de sus vecinos de dos saltos, elige de entre sus vecinos de un salto un grupo de nodos MPR que retransmitan sus mensajes, de forma que le proporcionen acceso hacia todos los vecinos de dos saltos; de esta forma abrirán la ruta hacia cualquier nodo de la red. Los nodos elegidos como MPR son notificados de su condición, manteniendo información sobre quiénes le han elegido como MPR (denominados selectores MPR) en una estructura llamada conjunto selector MPR.

Uno de los cometidos de los MPR es retransmitir los mensajes de difusión generados por alguno de los nodos en su conjunto selector MPR. De este modo, los mensajes llegan a toda la red, pero intentando que la saturación sea mínima.

La otra tarea que debe realizar un nodo que ha sido seleccionado como MPR es generar y retransmitir mensajes *Topology Control* (TC), que dan a conocer al resto de la red los nodos de los que el emisor tiene constancia. Los mensajes TC se generan de forma periódica, y contienen una lista con las direcciones de los nodos del conjunto selector MPR, nodos que han elegido al emisor del mensaje como MPR (a diferencia de otros protocolos que anunciarían a cualquier nodo cercano). Con esto se consigue que la información sobre la topología generada sea mínima, y que su diseminación por la red se haga de forma controlada.

xEn otras palabras, los MPR tienen también la función de anunciar información sobre la topología de la red mediante inundación controlada, para que todos los participantes conozcan la ruta hacia el resto de la red. Como se ha indicado anteriormente, la inundación se realiza mediante mensajes *Topology Control* (TC), generados por nodos que han sido seleccionados como MPR, y que se reenvían de forma eficiente entre los MPR en lugar de hacerse mediante difusión masiva.

También es importante conocer las estructuras de datos internas que maneja OLSR. En particular, la más relevante es la tabla de rutas. La información que se tiene de cada nodo de la red en la tabla de rutas es una entrada con los siguientes campos:

R_dest_addr ***R_next_addr*** ***R_dist*** ***R_iface_addr***

Esa entrada significa que el nodo identificado por la dirección *R_dest_addr* está a una distancia estimada de *R_dist* saltos, que el vecino con la dirección de interfaz *R_next_addr* es el siguiente salto en la ruta hacia *R_dest_addr*, y que este vecino es alcanzable a través de la interfaz local con la dirección *R_iface_addr*.

En OLSR se tiene en consideración la posibilidad de que un nodo tenga más de una interfaz de red participando al mismo momento en la red. Cada dirección de interfaz está asociada con una dirección principal, única para cada nodo. Esta dirección principal será la misma que la dirección de la interfaz en caso de que ese nodo tenga una única interfaz utilizando OLSR.

11.3.2 Formato del Paquete OLSR

Las comunicaciones en OLSR se realizan usando un formato de paquete común para toda la información relacionada con el protocolo. De este modo, se facilitan futuras ampliaciones del protocolo sin romper la compatibilidad con versiones anteriores. Además, esto también facilita agrupar diferentes tipos de información dentro de una misma transmisión.

Los paquetes se encapsulan dentro de datagramas UDP para su transmisión por la red.

Cada paquete encapsula a su vez uno o varios mensajes. Los mensajes comparten un formato de cabecera común, lo que permite que un nodo sea capaz de aceptar y retransmitir (si procede) mensajes de tipo desconocido.

Los mensajes pueden ser transmitidos por inundación a la red en su totalidad, o la transmisión puede ser limitada a nodos dentro de un determinado diámetro -refiriéndonos a número de saltos- desde el emisor del mensaje. Por lo tanto, transmitir un mensaje al vecindario de un nodo es un caso especial de transmisión por inundación. Cuando se transmiten mensajes de control, las retransmisiones duplicadas son eliminadas de forma local, ya que cada nodo guarda información sobre los mensajes de control que ya ha transmitido anteriormente.

Los paquetes en OLSR usan el puerto UDP 698, asignado por el *Internet Assigned Numbers Authority* (IANA).

Los campos de cualquier paquete OLSR (omitiendo las cabeceras IP y UDP) se indican en la Figura 11.2 .

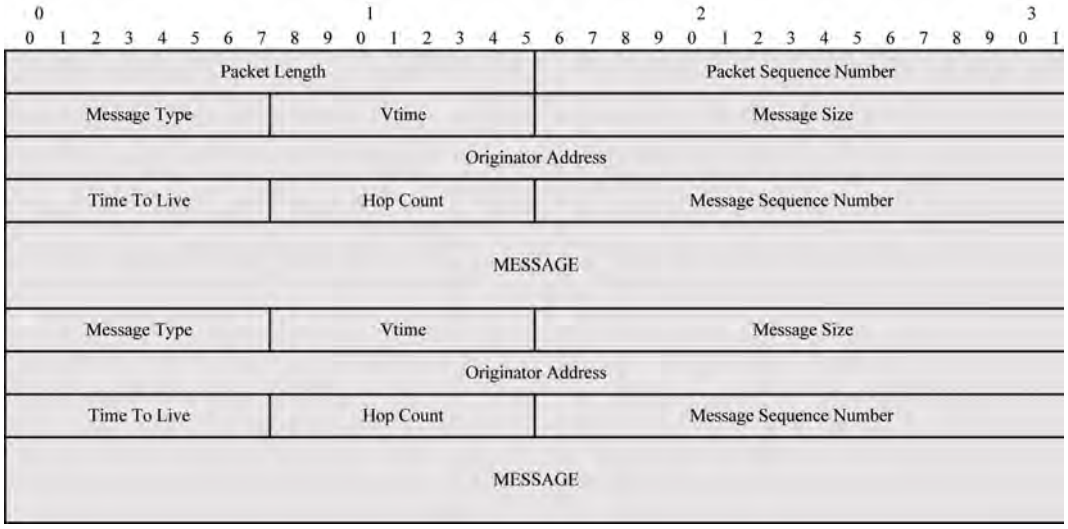


Figura 11.2: Formato del paquete OLSR

11.3.2.1 Cabecera del Paquete

La Figura 11.3 muestra los campos de la cabecera del paquete OLSR:

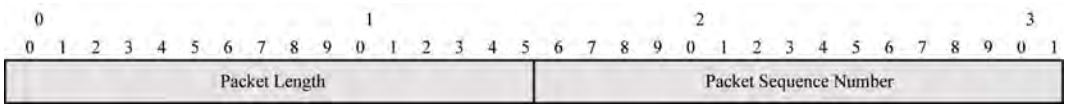


Figura 11.3: Cabecera del paquete OLSR

- *Packet Length*: [16 bits]. En este campo se define el tamaño del paquete OLSR en bytes.

- *Packet Sequence Number*: [16 bits]. Este campo se utiliza para definir el número de secuencia del paquete. Debe ser incrementado en uno cada vez que un nuevo paquete OLSR es transmitido.

11.3.2.2 Cabecera del Mensaje

En la Figura 11.4 se representan los campos de cabecera del mensaje:

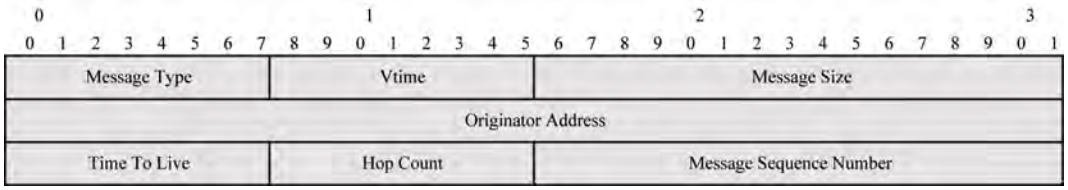


Figura 11.4: Cabecera del mensaje

- *Message Type*: [8 bits]. Este campo indica qué tipo de mensaje se encuentra en el campo MESSAGE. Los tipos en el rango 0-127 son reservados.

- *Vtime*: [8 bits]. Campo que indica el tiempo de validez de la información contenida en el mensaje.

- *Message Size*: [16 bits]. Tamaño del mensaje en bytes, incluyendo la cabecera y el campo MESSAGE.

- *Originator Address*: [32 bits]. Indica a dirección principal del nodo que originalmente generó el mensaje. No debe confundirse este campo con la dirección origen de la cabecera IP, que se modifica cada vez que el paquete OLSR se retransmite por un nodo intermedio.

- *Time To Live*: [8 bits]. Contiene el número máximo de saltos que el mensaje será transmitido. Antes de retransmitir un mensaje, al valor de este campo se le debe restar uno.

- *Hop Count*: [8 bits]. Número de saltos que el mensaje ha dado. Se inicializa a 0, y se incrementa en 1 en cada retransmisión.

- *Message Sequence Number*: [16 bits]. Al generar un mensaje, el nodo que lo genera le asigna un número de secuencia único que identifica a cada mensaje en este campo. El número de secuencia se incrementa en 1 para cada mensaje originado por el nodo

11.3.3 Mensaje MID

Si el nodo tiene más de una interface se anuncia esta interfaz adicional periódicamente a los otros nodos utilizando mensajes *Multiple Interface Declaration* (MID).

11.3.4 Mensaje Hello

El protocolo OLSR utiliza el intercambio periódico de mensajes **Hello** para descubrir a los nodos vecinos y el estado de la red a nivel de enlace. El formato del mensaje se indica en la Figura 11.5 . Como la dirección principal del nodo está incluida en la dirección origen del encabezado del mensaje solo las direcciones de las interfaces adicionales tienen que ser anunciadas. Con base en esta información se construye el *Multiple Interface Association Information Base* en el nodo receptor.

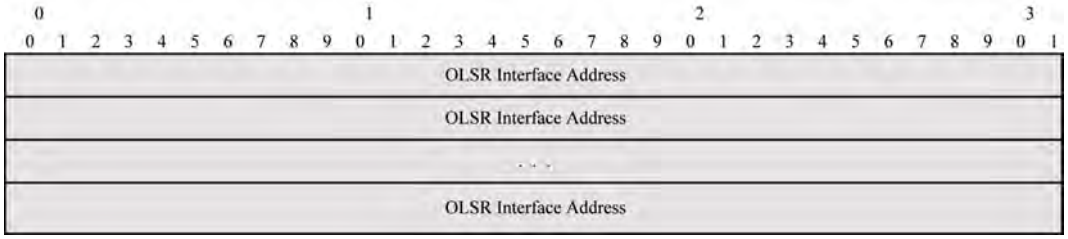


Figura 11.5: Formato del mensaje MID

11.3.4.1 Formato del Mensaje Hello

Los mensajes **Hello** siguen un formato similar al del paquete general, de modo que puedan incluir información para la detección del estado de los enlaces de la red, para transmitir señales para la detección de nodos vecinos, para selección de MPRs y para tener en cuenta futuras ampliaciones.

El formato del mensaje se indica en la Figura 11.6 .

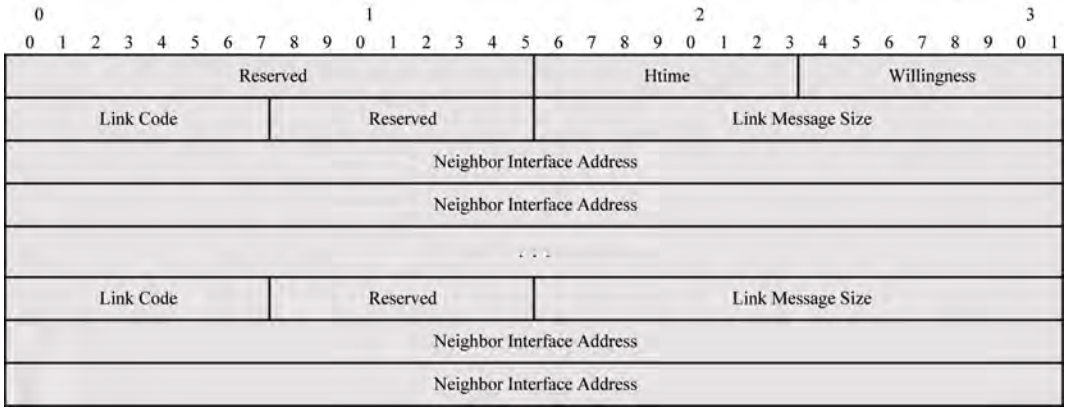


Figura 11.6: Formato del mensaje **Hello**

Se envía como datos dentro del paquete general OLSR descrito anteriormente, configurando el campo *Message Type* con el valor HELLO_MESSAGE y el campo TTL con 1.

- *Reserved*: [16 bits]. Reservado, se establece con el valor 0x0000.
- *HTime*: [8 bits]. Este campo especifica el intervalo de emisión de mensajes *Hello* usado por el nodo, esto es, el tiempo hasta el envío del próximo *Hello*.

- *Willingness*: [8 bits]. Indica la disponibilidad que tiene un nodo para re-enviar tráfico a otros nodos, si la disponibilidad de un nodo se define como WILL_NEVER, éste nunca será elegido como un nodo MPR.

- *Link Code*: [8 bits]. Este campo especifica el tipo de enlace que el nodo emisor tiene con los vecinos en su lista. Como mínimo, OLSR requiere los siguientes tres tipos de enlaces:

  - *ASYM_LINK*: Indica que los enlaces entre el nodo emisor y sus vecinos son de tipo asimétricos (es decir, solamente es posible "escuchar" al vecino, pero no es posible establecer un enlace bidireccional con éste).
  - *SYM_LINK*: Indica que el enlace entre el emisor y sus vecinos son simétricos (existe un enlace bidireccional).
  - *MPR_LINK*: Indica que los nodos definidos en la lista han sido seleccionados por el emisor como MPR.

- *Reserved*: [8 bits]. Este campo se reserva para uso futuro, y debe ser puesto a 0x00.

- *Link Message Size*: [16 bits]. Este campo define el tamaño del mensaje de enlace, el cual se mide desde el inicio del campo *Link Code* hasta el siguiente campo de *Link Code*. Si no existe otro campo de *Link Code* dentro del mensaje *Hello*, entonces el valor del campo *Link Code* se mide hasta el fin del mensaje *Hello*.

- *Neighbor Interface Address*: [32 bits]. Este campo define la lista de vecinos que se han etiquetado con un *Link Code* en particular.

11.3.4.2 Procesamiento del Mensaje *Hello*

Los nodos procesan los mensajes *Hello* recibidos para la detección de conexiones a nivel de enlace, detección de vecinos y selección de MPR.

11.3.5 Descubrimiento de Vecinos

11.3.5.1 Detección de Conexiones a Nivel de Enlace

Cada nodo guarda información sobre sus conexiones a nivel de enlace con otros nodos en una estructura llamada *Link Set*. Con estas conexiones nos referimos más concretamente a las interfaces de red utilizando OLSR y su capacidad de intercambiar paquetes OLSR.

El mecanismo usado para esta detección es el intercambio periódico de mensajes *Hello*. Para considerar una conexión válida se debe comprobar que existe comunicación (recepción de *Hello*) en ambos sentidos.

Cada nodo vecino tiene asociado un estado en relación a la conexión: *simétrico* o *asimétrico*. El primero indica que la comunicación en ambos sentidos ha sido confirmada; y el segundo se usa para indicar que se han recibido mensajes *Hello* generados por el nodo vecino, pero no se ha confirmado aún que el nodo vecino es capaz de recibir los *Hello* generados localmente.

La confirmación de que un nodo vecino es capaz de recibir los mensajes *Hello* emitidos se tiene al encontrar la dirección propia en los *Hello* del vecino.

11.3.5.2 Detección de Vecinos

Cada nodo mantiene un conjunto de tuplas de vecinos (*neighbor tuples*) basadas en la información sobre las conexiones almacenadas en el *Link Set*. Cada tupla de vecino consta de los datos:

N_neighbor_main_addr ***N_status*** ***N_willingness***

donde *N_neighbor_main_addr* es la dirección principal del nodo vecino, *N_status* se refiere al estado de la conexión (simétrica o asimétrica), y *N_willingness* es un entero entre 0 y 7 que representa la disposición o intención del vecino de retransmitir tráfico de otros nodos.

Aparte del conjunto de vecinos inmediatos o de un salto, con los que se tiene conexión a nivel de enlace, cada nodo guarda información sobre el conjunto de nodos de dos saltos de distancia en la estructura *2-hop Neigbor Set*. Para ello, por cada nodo vecino de un salto, se guarda el conjunto de sus vecinos de un salto, ya que están anunciados en los mensajes *Hello* que se reciben periódicamente.

11.3.6 Multipuntos de Retransmisión (MPR)

Los MPR (Multipoint Relay) se usan para inundar mensajes de control de un nodo a toda la red minimizando las retransmisiones. Por lo tanto, el concepto de MPR se considera una optimización del mecanismo habitual de inundación de mensajes en una red.

Como se ilustra en la Figura 11.7 , cada nodo en la red elige, independientemente de los demás, su propio conjunto de MPR de entre sus vecinos de un salto con conexión simétrica. El conjunto de nodos seleccionados se conoce como el conjunto MPR de ese nodo. Los vecinos del nodo N que no están dentro del grupo MPR, reciben y procesan la información de los mensajes de difusión, pero no retransmiten la información proveniente del nodo N.

La sobrecarga del tráfico de control generada por el protocolo de enrutamiento es directamente proporcional al tamaño del conjunto de nodos MPR en la red. A su vez, los nodos MPR mantienen información sobre el conjunto de vecinos a un salto que lo han seleccionado como MPR; este conjunto se conoce como conjunto selector de MPR de un nodo. Está información se adquiere de los mensajes *Hello* recibidos de los vecinos a un salto.

Aunque la inundación de mensajes pura es más confiable y robusta, ésta consume una gran cantidad del ancho de banda. El empleo de nodos MPR proporciona resultados igualmente buenos, con mucho menos tráfico de control. En la Figura 11.8 se ilustra una comparación, en términos de retransmisiones, para hacer llegar un mensaje de difusión a 3 saltos en la red.

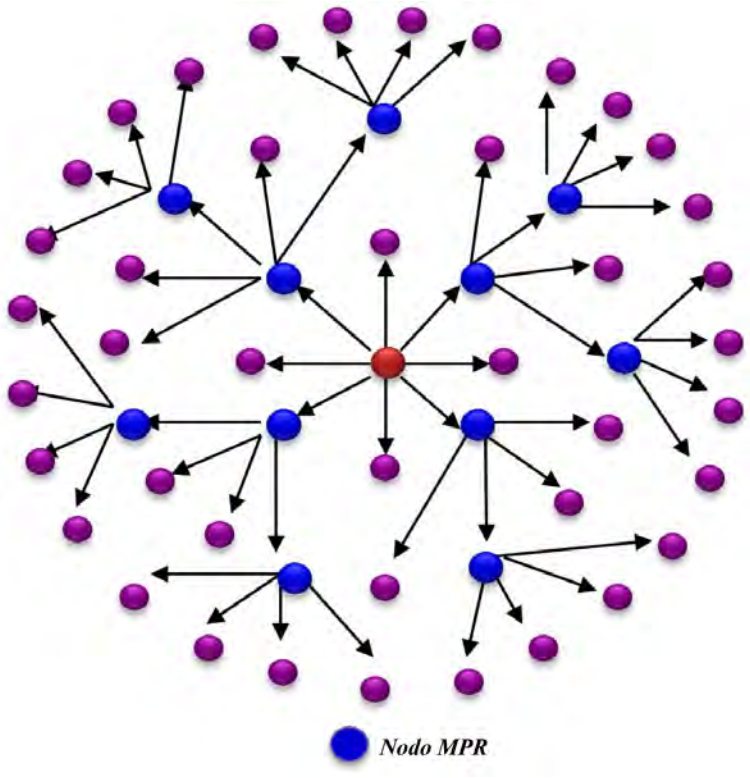
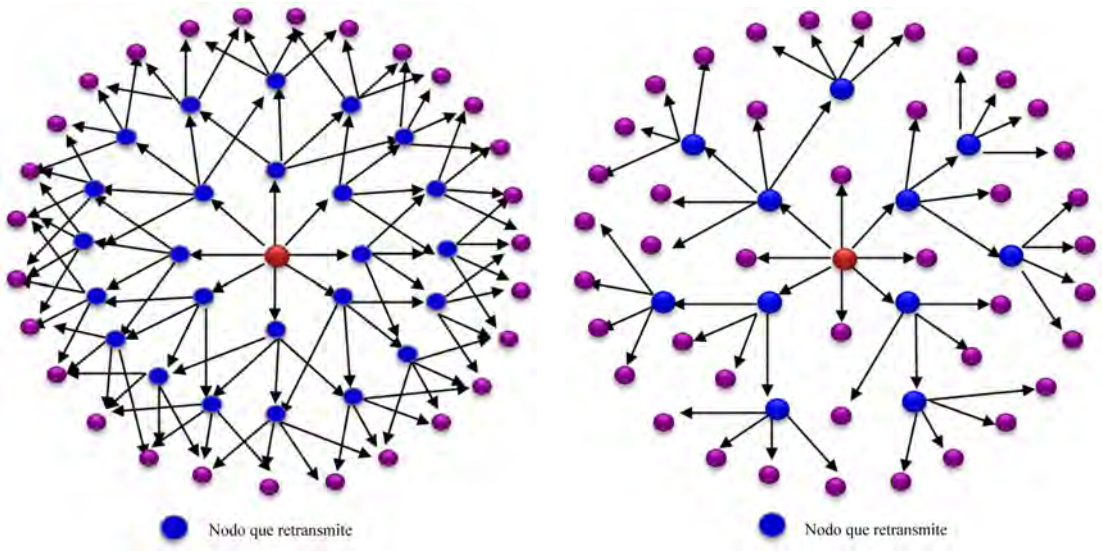


Figura 11.7: Proceso de selección de nodos MPR



(a) Difusión pura

(b) Difusión usando nodos MPR

Figura 11.8: Diferencia entre la difusión pura y la difusión usando nodos MPR

11.3.6.1 Selección de MPR

Cuando se ha elegido a un vecino como MPR, se anuncia en los mensajes *Hello* colocando el valor MPR_NEIGH en lugar de SYM_NEIGH en el campo *Link Type* anterior a

la dirección del vecino elegido como MPR.

El conjunto de MPR es calculado por cada nodo de forma que, a través de los vecinos elegidos, el nodo sea capaz de alcanzar a todos los vecinos de dos saltos. Más concretamente, a los vecinos de dos saltos exactos, por lo que no se está contando a los vecinos de un salto. Aunque el conjunto de MPR no debe ser mínimo para garantizar el correcto funcionamiento, cuanto más pequeño sea se consigue menor sobrecarga producida por los mensajes de control del protocolo OSLR.

Cada nodo guarda además en el conjunto selector MPR el conjunto de nodos que le han elegido como MPR. Son detectados al procesar los mensajes *Hello* recibidos.

11.3.7 Descubrimiento de la Topología en OLSR

11.3.7.1 Funcionamiento

La detección de conexiones y vecinos proporciona a cada nodo una lista de nodos con los que comunicarse directamente y, haciendo uso de nodos MPR, un mecanismo para inundaciones optimizado. Basándose en esto, se genera información sobre la topología y se distribuye por la red.

Los mensajes *Topology Control* (TC) los generan los nodos que han sido elegidos como MPR por algún vecino suyo. Sirven para anunciar un conjunto de enlaces entre el emisor y otros nodos, que por lo general será su conjunto selector MPR, es decir, los vecinos que han elegido al nodo emisor como MPR. Estos mensajes TC se emiten a toda la red por inundación.

Debido a limitaciones de tamaño de los mensajes en la red, la lista de direcciones anunciadas puede ser parcial en cada mensaje TC. Sin embargo, al unir todos los mensajes TC emitidos deben encontrarse todas las direcciones del conjunto selector MPR.

11.3.7.2 Formato de los Mensajes TC

Los mensajes TC tienen el siguiente formato mostrado en la Figura 11.9 :

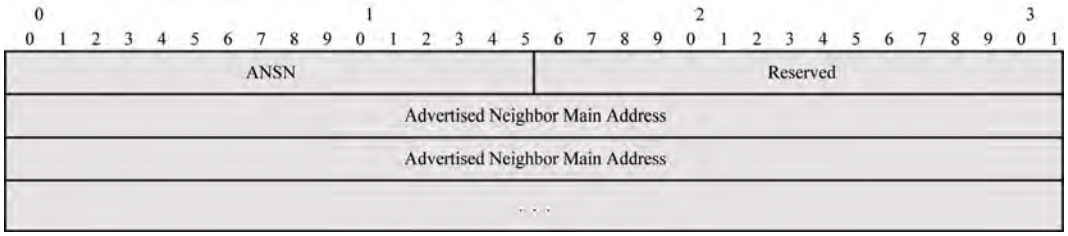


Figura 11.9: Formato del mensaje TC

Esto se envía como datos dentro del paquete OLSR, configurando el campo *Message Type* con el valor TC_MESSAGE y el campo TTL con el valor 255 (el máximo).

- *Advertised Neighbor Sequence Number (ANSN)*: [16 bits]. Un número de secuencia que se asocia con el conjunto de vecinos que se anuncia en este mensaje. Cada vez que un nodo detecta cambios en el conjunto de sus vecinos, se incrementa este número. Sirve para que los nodos que reciben este mensaje sepan si se trata de información más reciente que la que puedan tener actualmente.

- *Reserved*: [16 bits]. Este campo está reservado. Se le da el valor 0x0000.

- *Advertised Neighbor Main Address*: [32 bits]. Campo que contiene la dirección principal de un nodo vecino.

11.3.8 Cálculo de las Tablas de Rutas

Cada nodo mantiene actualizada una tabla con rutas hacia el resto de nodos de la red. Esta tabla se basa en la información obtenida sobre los nodos vecinos y los mensajes de control TC.

El formato de las entradas en esta tabla es el siguiente:

R_dest_addr	***R_next_addr***	***R_dist***	***R_iface_addr***
R_dest_addr	***R_next_addr***	***R_dist***	***R_iface_addr***

...

Cada entrada significa que el nodo con la dirección *R_dest_addr* se encuentra a una distancia de *R_dist* saltos del nodo local, que el nodo vecino con la dirección de interfaz de red *R_next_addr* es el siguiente salto en la ruta hacia *R_dest_addr*, y que este nodo es alcanzable desde la interfaz local con la dirección *R_iface_addr*. Se mantiene una entrada por cada nodo de la red para el que se tiene ruta conocida.

Los nodos con una ruta desconocida no se incluyen. La actualización de la tabla se realiza en caso de aparición o desaparición de un nodo, ya sea vecino inmediato, vecino de dos saltos de distancia, o cualquier otro nodo conocido a través de los mensajes de control TC. También se actualiza la tabla cuando cambia información sobre las interfaces múltiples que puedan estar asociadas a los nodos.

Esta actualización de la tabla es un proceso interno, que no desencadena el envío de ningún mensaje.

11.4 Protocolo AODV

El protocolo de encaminamiento *Ad hoc On-demand Distance Vector* (AODV) [PBRD03] es un protocolo bajo demanda basado en encaminamiento por vector distancia. Los nodos que no tienen ninguna ruta activa no almacenan información de encaminamiento ni participan en el intercambio de tablas de encaminamiento. Un nodo no tiene, por tanto, que descubrir ni guardar información de una ruta hacia otro nodo hasta que no se comunique con él, salvo que sea nodo intermedio de dos nodos que tengan establecida una comunicación. Cada nodo mantiene una tabla de encaminamiento con la información que posee de las rutas, por lo que no es necesario que los paquetes lleven información de la ruta a seguir, con el consecuente ahorro en ancho de banda.

La tabla de encaminamiento tiene un tiempo de vida para cada entrada, de forma que si este tiempo expira reinicia la búsqueda de una ruta para el destino que tuviera asociado. De igual manera, las entradas de la tabla llevan asociadas un número de secuencia que sirve para evitar bucles en las rutas, además de ayudar a distinguir información antigua de información actualizada posteriormente. El correcto funcionamiento de AODV depende principalmente de que cada nodo mantenga actualizado su propio número de secuencia.

11.4.1 Mensajes de Control

Seguidamente se describen los mensajes de control definidos en la especificación de AODV, conocidos como mensajes genéricos de AODV.

11.4.1.1 Mensajes RREQ

Los mensajes de petición de ruta o RREQ (*Route REQuest*) son utilizados por los nodos para comenzar el proceso de descubrimiento de ruta cuando desean comunicarse con otro nodo. Para ello, primero aumentan su propio número de secuencia y a continuación inundan la red con un mensaje RREQ. El formato de este mensaje se indica en la Figura 11.10 . Durante un tiempo el nodo guardará el identificador del mensaje y la dirección origen del mismo para evitar procesarlo si le llega de vuelta (véase Figura 11.11).

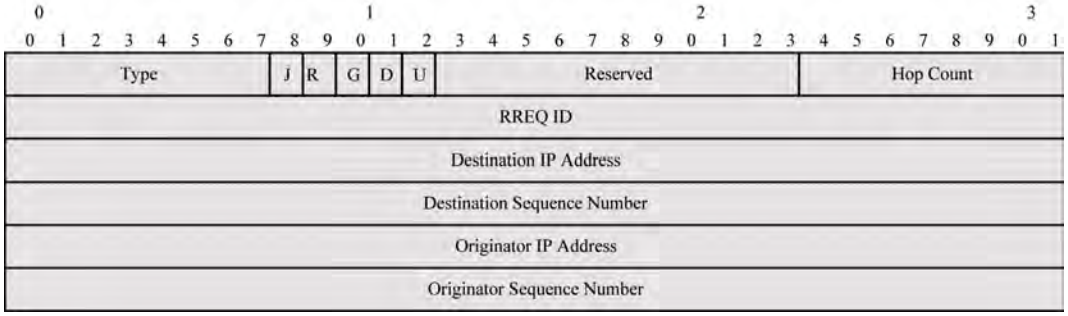


Figura 11.10: Formato del mensaje RREQ de AODV

Para intentar ahorrar ancho de banda se usa la *técnica del aumento del anillo de búsqueda* que consiste en que el RREQ enviado tiene un tiempo de vida o TTL (*Time To Live*) mínimo. De esta manera, sólo los dispositivos cercanos al nodo origen reciben el RREQ. Si no se obtiene respuesta se va aumentando el área donde se reciben los mensajes RREQ mediante el incremento del TTL, hasta llegar a un límite de TTL. Esta técnica limita la propagación de los mensajes de RREQ y, en el caso de no obtener respuesta, va aumentando el alcance de los mensajes.

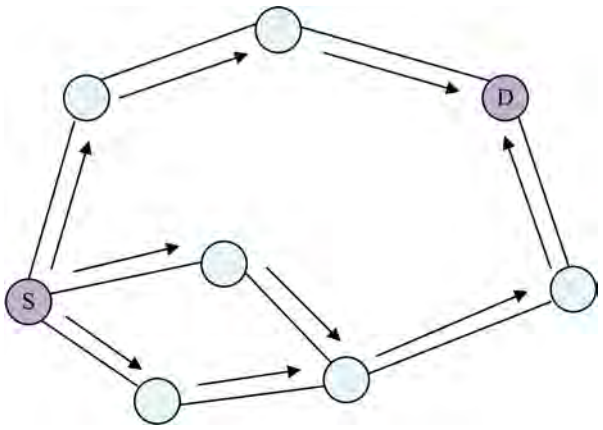


Figura 11.11: Propagación del un mensaje RREQ en AODV

Un nodo que recibe un RREQ debe crear o actualizar una ruta hacia el nodo vecino que se lo ha transmitido. Después comprueba si es un mensaje duplicado y si es así no realiza ningún proceso más. Si no es duplicado el nodo crea o actualiza una ruta inversa hacia el origen del mensaje RREQ. Si ya existe dicha ruta se tiene que actualizar su número de secuencia con el del mensaje RREQ si éste último es mayor. El "siguiente salto" será el

vecino del que se ha recibido el mensaje. Por esta ruta se puede retransmitir el RREP si viene de vuelta.

Si el nodo que recibe el RREQ no está en condiciones de generar un RREP debe retransmitir el RREQ, actualizando antes el número de secuencia para el destino del mensaje con el suyo propio si es mayor que el que lleva el mensaje.

11.4.1.2 Mensajes RREP

Los mensajes de respuesta de ruta o RREP (*Route REPly*) se envían como respuesta a la llegada de un RREQ, si el nodo es el destino o si tiene información actualizada para llegar a él. Se detecta que la información está actualizada gracias a los números de secuencia. El formato de los mensajes RREP puede verse en la Figura 11.12 .

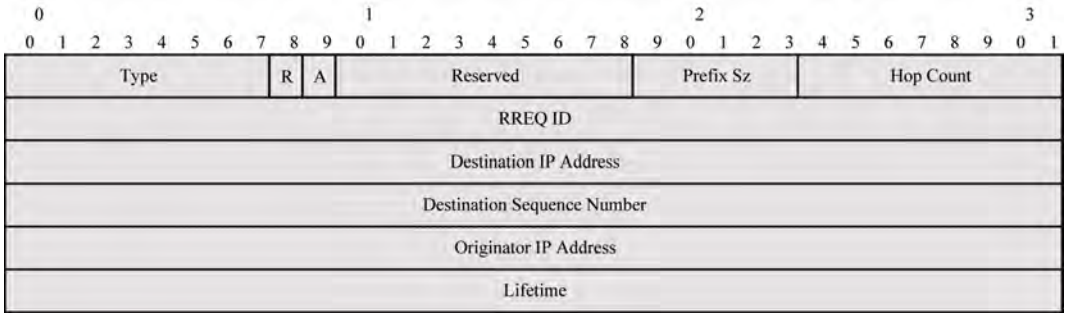


Figura 11.12: Formato del mensaje RREP de AODV

Los mensajes RREP no inundan la red sino que se envían en *unicast* hacia el nodo que originó el proceso de descubrimiento de ruta por el camino inverso creado con la inundación del RREQ (Figura 11.13).

Si el nodo que genera el RREP es el destino, justo antes de enviarlo, incrementa en una unidad su número de secuencia si ese es el valor anunciado por el RREQ. Si es un nodo intermedio el que lanza el RREP pone en el mensaje el número de secuencia que posee para el destino.

Los nodos que procesan un RREP crean o actualizan la ruta hacia el vecino que se lo ha enviado. Además, crean o actualizan la ruta directa hacia el destino (el emisor del RREP) para poder encaminar los paquetes que vayan destinados a ese nodo.

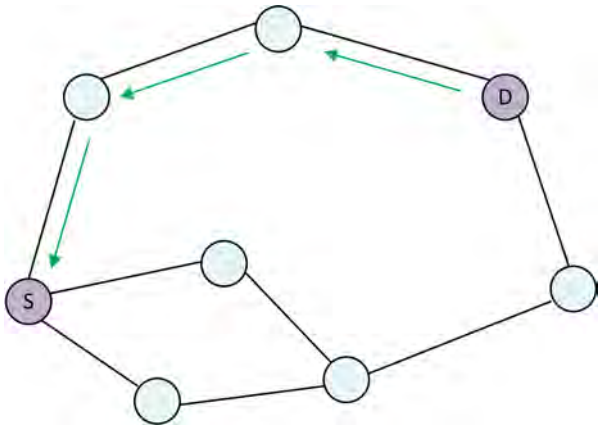


Figura 11.13: Camino del mensaje de vuelta RREP al origen

11.4.1.3 Mensajes RERR

Los mensajes RERR (*Route ERRor*) se utilizan para notificar que no se puede alcanzar un destino determinado. El formato de un mensaje RERR puede observarse en la Figura 11.14 .

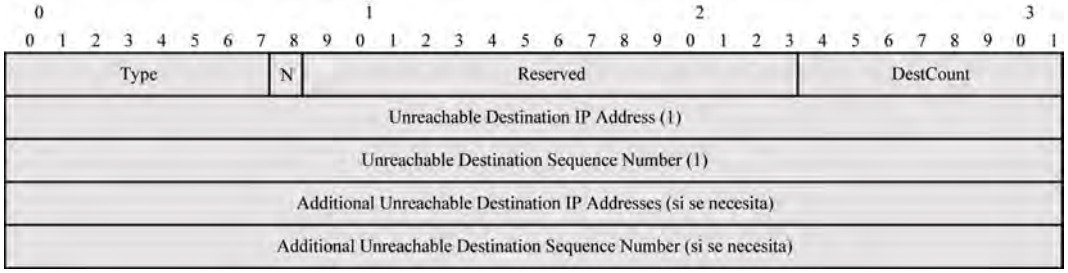


Figura 11.14: Formato del mensaje RERR de AODV

El que un nodo no sea capaz de alcanzar un determinado destino puede deberse a tres situaciones distintas:

- Cuando un nodo detecta la pérdida de conectividad con un vecino que es el "siguiente salto" de una ruta activa.
- Cuando un nodo tiene que enviar un paquete dirigido a un destino del que no se conoce ninguna ruta activa.
- Cuando un nodo recibe un RERR de un vecino anunciando la pérdida de conectividad con vecinos que utilizaba con "siguiente salto" en rutas activas.

11.4.1.4 Mensajes *Hello*

Los nodos que forman parte de rutas activas pueden enviar información de conectividad a sus vecinos mediante mensajes *Hello*. Se generan cuando en un determinado período de tiempo no han transmitido ningún mensaje *broadcast*.

Los mensajes *Hello* son realmente RREP con un TTL o tiempo de vida de un salto para que sólo sean recibidos por los vecinos del nodo que los envía. Su formato puede

observarse en la Figura 11.12 . Cuando un vecino procesa un mensaje *Hello* debe crear o actualizar la entrada de la tabla de encaminamiento cuyo destino es el origen del mensaje. Si algún vecino recibe un mensaje *Hello* de un nodo, y tras un período de tiempo de espera no recibe ningún otro mensaje de él, da el enlace por perdido.

11.4.2 Descubrimiento de Rutas

El descubrimiento de rutas se hace inundando la red con mensajes RREQ. Cuando un nodo recibe un RREQ crea o actualiza una entrada en su tabla de encaminamiento hacia el origen de la petición, estableciendo de esta forma un camino inverso hacia el nodo origen al presuponer que los enlaces son simétricos. Cuando la petición llega al nodo destino genera un mensaje de respuesta RREP que transmite al nodo origen por el camino inverso establecido.

Un nodo intermedio también puede enviar un RREP si conoce un camino más reciente. Para ello se usan números de secuencia de destino. A cada nuevo RREQ desde la fuente hacia un destino se le asigna un número mayor. De esta forma, si un nodo intermedio conoce una ruta pero tiene un número menor, no envía el RREP. Además de para evitar utilizar rutas antiguas o rotas, los números de secuencia sirven para prevenir la formación de bucles que degraden la eficiencia de la red.

Mediante el RREP de vuelta hacia el emisor por el camino inverso, se establece la ruta entre los nodos origen y destino. A su vez, el envío de RREP se utiliza para que los nodos intermedios por los que el mensaje va pasando actualicen sus tablas de encaminamiento.

Una optimización para este procedimiento de descubrimiento de ruta es la técnica del aumento del anillo de búsqueda ya comentada anteriormente. Consiste en enviar los mensajes RREQ con un tiempo de vida (TTL) bajo para evitar su propagación por toda la red. Si tras un tiempo no se ha recibido el RREP se envía otro RREQ con un TTL mayor. Este proceso de incrementar el TTL podrá repetirse hasta que se alcance un TTL umbral. Una vez superado, se inunda la red.

11.4.3 Mantenimiento de Rutas

En la tabla de encaminamiento de AODV se distinguen las entradas en función de si fueron creadas al recibir un RREQ o un RREP. Si se crearon con la llegada de un mensaje RREP son rutas hacia delante, las cuales se eliminan si no se usan durante un intervalo de tiempo de ruta activa, es decir, si no se transmite ningún dato por esa ruta, aunque la ruta siga siendo válida. Si la ruta se conoció por un mensaje RREQ se dice que la ruta es hacia atrás y se elimina transcurrido un intervalo de tiempo, normalmente menor que el de ruta activa y suficientemente amplio como para permitir la vuelta del RREP.

Cuando el enlace al siguiente salto en una entrada de tabla se rompe, se informa a todos los vecinos activos. Un vecino de un nodo se considera activo para una entrada si envió un paquete por dicha entrada dentro del intervalo de ruta activa. Las rupturas de enlace se propagan por medio de mensajes de error de ruta, o RERR, que también actualizan los números de secuencia de destino.

Cuando un nodo no puede transmitir un paquete por fallo del siguiente enlace, incrementa su número de secuencia de destino y genera un RERR que incluye dicho número. Cuando la fuente recibe el RERR inicia un nuevo descubrimiento de ruta hacia el destino anterior, pero usando un número de secuencia al menos tan grande como el recibido. Al llegar el nuevo RREQ con el número dado al nodo destino, éste lo establece como su número de secuencia, salvo que ya tenga un número mayor que el recibido.

Los nodos vecinos pueden intercambiarse periódicamente mensajes *Hello* para detectar los fallos de enlace. La no recepción de este tipo de paquetes de un vecino activo puede interpretarse como la ruptura del enlace entre ellos.

11.5 ZRP

Zone Routing Protocol (ZRP) [HPS02] es un protocolo de encaminamiento híbrido, ya que combina las mejores propiedades de los protocolos proactivos y reactivos.

ZRP se basa en separar la red en zonas. Se diferencian claramente una zona cercana o vecindario, compuesta por los nodos que están a un máximo de N saltos, y el resto de nodos de la red.

En ZRP se usan dos componentes para el encaminamiento: *IntrA-zone Routing Protocol* (IARP) e *IntEr-zone Routing Protocol* (IERP).

En la zona cercana se usa el componente *IntrA-zone Routing Protocol* (IARP) que actúa como un protocolo proactivo, manteniendo todas las rutas de los nodos que se encuentren a N saltos o menos, siendo N variable. El mecanismo usado para descubrir los nodos vecinos es el intercambio periódico de mensajes *Hello*.

ZRP cuenta con el componente *IntEr-zone Routing Protocol* (IERP) para el encaminamiento global hacia los nodos fuera de la zona interior o cercana, que se comporta como un protocolo reactivo.

Cuando se necesita la ruta hacia un nuevo nodo se usa el componente IERP, y se pide esta ruta con el mecanismo *Bordercast Resolution Protocol* (BRP). Este mecanismo funciona enviando mensajes de petición de ruta a los nodos que pertenecen a la frontera o borde de la zona interior con la zona exterior, es decir, a los nodos que están al número máximo de saltos de la zona interior.

Si alguno de estos nodos del borde conoce la ruta, envía un mensaje indicándosela al nodo que originó la petición. Si no es así, reenvían la petición por toda la red hasta que llegue a un nodo que conozca una ruta hacia el destino. Entonces se envía la respuesta de vuelta hasta el origen, guardando los nodos intermedios por los que pasa el mensaje para poder ser usados como ruta hacia el destino buscado.

Como se ha mencionado, el radio (en número de saltos) de la zona interior es ajustable según las necesidades de la red. Como casos extremos tenemos que si este radio es pequeño, como mínimo uno, ZRP se comportará como un protocolo puramente reactivo. Si por el contrario el radio es infinito, el comportamiento será proactivo.

11.6 Resumen

El principal objetivo de este capítulo ha sido mostrar que el envío de un mensaje de un nodo a otro sin existir un enlace directo constituye la finalidad de los protocolos de encaminamiento para redes móviles ad hoc. Asimismo, se han presentado diversas clasificaciones de los mismos siendo la más relevante aquella que los agrupa según el procedimiento adoptado para el descubrimiento del camino a establecer y a su mantenimiento y que divide a los protocolos de encaminamiento en tres clases: proactivos (protocolos en los que cada nodo mantiene información de cómo llegar a cualquier otro nodo de la red e intercambia esta información con todos sus vecinos) con baja latencia y alta sobrecarga, reactivos (protocolos en los que un nodo sólo calcula la ruta a un destino cuando es necesario un intercambio de paquetes con el mismo) con alta latencia y baja sobrecarga e híbridos (protocolos que combinan aspectos de los dos anteriores, siendo proactivos a nivel local y reactivos a nivel global) que minimizan los inconvenientes de ambos pero a costa de

aumentar la complejidad. Posteriormente, se ha visto un ejemplo de protocolo proactivo (OLSR), reactivo (AODV) e híbrido (ZRP), describiendo en detalle los dos primeros ya que se utilizan de patrones de referencia para analizar las prestaciones de los protocolos de encaminamiento especificados en la presente memoria.

Capítulo 12

El Algoritmo Ant Colony Optimization (ACO)

El objetivo general de este capítulo es presentar el algoritmo de optimización de la colonia de hormigas, la teoría sobre la que se sustenta los protocolos de encaminamiento desarrollados en la presente Tesis. En primer lugar se dan algunas nociones básicas del mismo. Luego se analiza el denominado experimento del doble puente. Seguidamente, se introducen las hormigas artificiales como agentes computacionales que tienen un comportamiento muy similar al de las hormigas naturales. A continuación, se describe la variante S-ACO, que encuentra el camino más corto en un grafo. Posteriormente, se comentan algunas generalidades de funcionamiento así como diversas técnicas de paralelización de la meta-heurística ACO. El capítulo finaliza con una breve síntesis de lo expuesto en el mismo.

12.1 Introducción

El algoritmo de optimización de la colonia de hormigas, tradicionalmente conocido por sus siglas en inglés, ACO, se compone de un conjunto de métodos y técnicas que se aplican en problemas genéricos de optimización.

El algoritmo ACO forma parte de los denominados algoritmos bioinspirados, y dentro de éstos, de aquellos basados en el concepto de inteligencia colectiva, que aplica el comportamiento social de los insectos y de otros animales para resolver problemas.

Mención especial merece el colectivo de las hormigas. La hormiga como individuo único tiene una efectividad limitada, pero como parte integrante de una colonia bien organizada, se convierte en un agente poderoso que trabaja para el desarrollo de la colonia. Las hormigas viven para la colonia y existen sólo como parte de ella. Las hormigas tienen la posibilidad de comunicarse, de aprender, de cooperar, de organizarse, etc., y todas juntas pueden llevar a cabo una misión concreta.

Existe un número considerable de investigadores que han estudiado el comportamiento de las hormigas en detalle. Uno de los patrones de comportamiento que más sorprende de las hormigas es la habilidad de ciertas especies para encontrar la comida por el camino más corto. Los biólogos han demostrado experimentalmente que se comunican por medio de una sustancia química denominada feromona.

Las hormigas, en su necesidad de encontrar la comida y traerla de vuelta al hormiguero, exploran una extensa área y se lo indican a otras al hacer el recorrido de vuelta a la colonia. De esta forma, las hormigas conocen el camino desde su hormiguero hasta su destino, sin necesidad de tener una visión global del terreno. La mayoría de las veces encuentran el

camino más corto y se adaptan a los cambios del terreno, demostrando su eficiencia en esta tarea.

Este patrón de comportamiento inspiró a los investigadores a desarrollar algoritmos de optimización que ayudasen a superar los diferentes problemas de encaminamiento existentes. Los primeros intentos aparecieron cerca de los noventa y han seguido evolucionando hasta nuestros días.

12.2 Experimento del Doble Puente

El experimento del doble puente [GADP89] consiste en observar cómo se comporta una colonia de hormigas de la especie argentina Linepithema humile (conocida como Iridomyrmex humilis hasta 1992) ante el problema de encontrar una fuente de comida.

Estas hormigas, que están completamente ciegas, realizan la comunicación entre ellas y su entorno por medio de una sustancia química llamada feromona. La señal que dejan en el suelo suele denominarse pista de feromona y, lógicamente, es fundamental para su vida social. Este tipo de feromona es usada por algunas especies de hormigas para marcar la ruta sobre la tierra y de esta forma conocer el camino hacia la comida y el de vuelta al hormiguero. Las hormigas huelen la feromona y eligen la ruta con mayor concentración de feromona.

Se ha comprobado que hay dos tipos de comportamientos: *asentamiento de la pista de feromona y seguimiento* de la misma. En el citado experimento se analiza el comportamiento de las hormigas que buscan comida en su trayectoria desde un punto a (el hormiguero) a otro punto b (la comida). En el experimento se cambia 3 veces la relación $r = l_l/l_s$ de la longitud de los ramales del doble puente, donde l_l es la longitud de rama más larga y l_s la longitud de la más corta.

- *Primer Caso*: El puente tiene dos ramales de igual longitud ($r = 1$). Las hormigas salen del hormiguero y empiezan a moverse libremente buscando un camino que les lleve hasta la comida. Se encuentran en el camino con dos ramales y un porcentaje de ellas elige un ramal y otro porcentaje elige el otro ramal. Se las observó durante un tiempo. El resultado fue que, aunque en la fase inicial la elección fue aleatoria, siguieron eligiendo los ramales en un porcentaje parecido.

- *Segundo Caso*: La relación de longitud entre los dos ramales es 2 ($r = 2$). De esta forma, el ramal largo es el doble que el corto. Se comprobó que, en un principio, la elección de camino fue aleatoria, como en el caso anterior, pero después de algún tiempo las hormigas eligieron el ramal más corto. Esto tiene la siguiente explicación: cuando comienza el experimento no hay feromona en ninguno de los dos ramales. Las hormigas no tienen una preferencia y eligen los dos ramales con la misma probabilidad. Como las hormigas depositan feromona mientras caminan, el camino más corto será recorrido más veces, por lo que al cabo de un tiempo la cantidad de feromona depositada en ese recorrido será mayor, y como eligen el que más feromona tiene, que es el más corto, irán por él ya que la mayor cantidad de feromona estimula a más hormigas a seguir ese camino. Sin embargo, no todas las hormigas usan el ramal corto, un pequeño número sigue por el ramal más largo. Esto se puede interpretar como un *mantenimiento de ruta*.

- *Tercer Caso*: Se estudió el comportamiento de las hormigas cuando se las presenta un solo camino y una vez que se acostumbran a él, se les ofreció otro camino más corto. Es decir, inicialmente se les presentó sólo un camino, que fue recorrido por

las hormigas durante 30 minutos, y después se les añadió un ramal que les hacía el recorrido más corto. La rama corta fue recorrida esporádicamente, pero la colonia siguió en la rama larga. Y es que la alta concentración de feromona en la rama más larga se mantenía por su lenta evaporación. Este comportamiento seguía reforzando el ramal más largo, a pesar de que ya tenían un ramal más corto.

12.3 Hormigas Artificiales

Las hormigas artificiales son agentes que colaboran para resolver problemas computacionales. En nuestro problema concreto la hormiga artificial es un agente computacional simple, que intenta dar soluciones al problema de cálculo del camino mínimo, explotando los rastros de feromona disponibles y la información heurística. En algunos casos ofrece soluciones que no son adecuadas y son *penalizadas*, descartándose o no dependiendo del nivel de error de la solución.

En general, la hormiga artificial tiene las siguientes propiedades:

- Busca solucionar el problema del *coste mínimo*.
- Tiene una memoria interna que almacena información sobre el camino seguido hasta el momento. Esta memoria sirve para:
  - (i) Construir soluciones válidas.
  - (ii) Evaluar la solución generada.
  - (iii) Reconstruir el camino que ha seguido la hormiga.
- Tiene un estado inicial con una o más condiciones de parada (estados finales).
- Comienza en el estado inicial y se mueve construyendo incrementalmente la solución.
- Cuando un nodo está en un estado determinado y ha seguido la secuencia de nodos visitados puede moverse a cualquier vecino.
- El movimiento se lleva a cabo aplicando una regla de transición, que está en función de la feromona depositada, de los valores heurísticos de la memoria interna de la hormiga y de las restricciones del problema.
- Cuando una hormiga se mueve de un nodo a otro puede actualizar el rastro de feromona asociado al *arco* entre los dos nodos. Este proceso se llama *actualización en línea de los rastros de feromona paso a paso*.
- El proceso de construcción acaba cuando se satisface alguna condición de parada.
- Una vez que la hormiga tiene una solución puede reconstruir el camino recorrido y actualizar los rastros de feromona de los arcos/componentes visitados/as utilizando un proceso llamado *actualización en línea a posteriori*.

Las colonias de hormigas naturales y artificiales comparten una serie de características, puesto que interaccionan y colaboran para solucionar una tarea determinada. Seguidamente se resumen las más importantes:

- Tanto las hormigas naturales como las artificiales modifican su entorno a través de una comunicación *estigmérgica*[1] basada en la feromona. En el caso de las hormigas artificiales, los rastros artificiales de feromona son valores numéricos que están disponibles únicamente de manera local.
- Las hormigas naturales y las artificiales comparten una tarea común: la búsqueda del camino más corto (construcción iterativa de una solución de coste mínimo) desde un origen (el hormiguero) hasta un estado final (la comida).
- Las hormigas artificiales construyen las soluciones de forma reiterada aplicando una estrategia de transición local estocástica o probabilística para moverse entre estados adyacentes, al igual que hacen las hormigas naturales.

Sin embargo, estas características por sí solas no permiten desarrollar algoritmos eficientes para problemas combinatorios difíciles. Las hormigas artificiales tienen algunas capacidades adicionales:

- Las hormigas artificiales pueden utilizar la información de forma heurística para encontrar el camino que les lleve al destino.
- Tienen una memoria que almacena el camino seguido.
- La cantidad de feromona depositada por la hormiga artificial está en función de la calidad de la solución encontrada.
- Otra gran diferencia está en el momento de depositar la feromona. Normalmente, las hormigas artificiales sólo depositan feromona después de generar una solución completa.
- La evaporación de feromona en los algoritmos ACO es diferente a como se presenta en la naturaleza, ya que la inclusión del mecanismo de evaporación es una cuestión fundamental para evitar que el algoritmo se quede estancado en la primera solución. La evaporación de feromona permite a la colonia de hormigas artificiales olvidar lentamente su historia y buscar nuevos caminos. Esto evita quedarnos en una solución prematura.
- Para mejorar la eficiencia y eficacia del sistema, los algoritmos ACO se enriquecen con habilidades adicionales como la capacidad de mirar más allá de la siguiente transición (*lookahead*), la optimización local, el *backtracking* y la llamada lista de candidatos, que contiene un conjunto de vecinos que pueden mejorar la eficiencia del algoritmo.

12.4 Simple Ant Colony Optimization (S-ACO)

A partir del experimento del doble puente Dorigo [Dor92] recrea el comportamiento de las hormigas para hacer un algoritmo que encuentra el camino más corto en un grafo.

Comienza considerando un grafo estático $G = (N, A)$, donde N es el conjunto de vértices del grafo y A el conjunto de aristas no dirigidas que los conecta. A los dos puntos entre los que se quiere establecer el camino más corto los llama fuente y destino, o como pasa con las hormigas reales, hormiguero y alimento.

[1] Colaboración a través del medio físico

El primer problema que aparece como consecuencia de la actualización de la feromona es la creación de ciclos. Puede acontecer mientras las hormigas construyen la solución ya que, como tienden a ir donde más feromona hay, repiten el camino que les parece mejor. Incluso si las hormigas escapan de los ciclos ya no se vería favorecida la ruta más corta entre la fuente y el destino. Por tanto, parece razonable eliminar la actualización de feromona a medida que la hormiga avanza. Sin embargo, si se suprime este mecanismo, el sistema deja de funcionar, incluso para el caso más simple del doble puente. La razón es muy simple: volviendo a las hormigas reales, la orientación de una hormiga se basa en el rastro que dejan en el suelo, de tal manera que cuando encuentra una fuente de comida debe volver al hormiguero. Si no marca su rastro con feromona no es capaz de volver, porque no recuerda el camino que ha seguido para alcanzar la comida.

En S-ACO, Dorigo extiende las capacidades básicas de las hormigas artificiales dotándolas de una memoria interna capaz de almacenar la ruta que han seguido así como el coste de la misma. Gracias a esta memoria las hormigas artificiales son capaces de implementar una serie de comportamientos que les permiten construir eficientemente una solución:

- A medida que la hormiga avanza, construye probabilísticamente una solución basada en rutas de feromona, sin mecanismo de actualización de feromona.
- Hace un recorrido inverso determinista con eliminación de ciclos, a la vez que se actualizan las rutas de feromonas.
- Evalúa la calidad de la solución generada y, en base a la misma, determina la cantidad de feromona que deposita.

12.4.1 Modos de funcionamiento

Las hormigas artificiales del S-ACO pueden funcionar de dos modos: *hacia adelante* (*forward*) y *hacia atrás* (*backward*). Se encuentran en modo *hacia adelante* cuando se mueven desde el hormiguero hacia la fuente de comida y en modo *hacia atrás* cuando regresan al hormiguero.

Cuando una hormiga en modo *hacia adelante* alcanza la fuente de comida cambia su comportamiento al modo *hacia atrás* y comienza su viaje de regreso hacia la colonia. En S-ACO las hormigas *hacia adelante* construyen una solución eligiendo de manera probabilística el siguiente nodo vecino al que se va a mover. La elección probabilística está influenciada por la feromona depositada previamente por otras hormigas en ese arco. Las hormigas artificiales en modo *hacia adelante* no depositan ninguna cantidad de feromona mientras se mueven. Este hecho, junto con el comportamiento determinista de las hormigas en modo *hacia atrás*, evita la aparición de ciclos.

12.4.2 Búsqueda de Caminos

Cada hormiga colabora en la solución del problema, comenzando desde el nodo fuente y tomando decisiones en cada nodo. La información que cada nodo almacena sobre sus vecinos es percibida por la hormiga y utilizada de una manera probabilística para decidir a qué siguiente nodo debe ir.

Al comienzo de cada proceso de búsqueda se asigna una cantidad constante de feromona a todos los arcos. Cuando la hormiga k se encuentra en el nodo i, utiliza el sendero de feromona τ_0 para calcular la probabilidad de elegir el nodo j de la siguiente manera:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha}{\sum_{l \in N_i^k} \tau_{il}^\alpha} & \text{si } j \in N_i^k \\ 0 & \text{si } j \notin N_i^k \end{cases} \tag{12.1}$$

donde N_i^k es la lista de los nodos disponibles a los que puede ir la hormiga k cuando se encuentra en el nodo i.

En S-ACO un nodo contiene la vecindad de todos los nodos que están conectados con él, exceptuando el nodo del que procede la hormiga. De esta manera se evita que las hormigas vuelvan a su nodo origen. Sólo en el caso de que la vecindad de un nodo sea el conjunto vacío, se permite a la hormiga volver sobre sus pasos. Hay que señalar que este proceder puede inducir fácilmente la generación de caminos cíclicos en el grafo.

12.4.3 Trazado de Ruta y Actualización de Feromona

El uso de una memoria explícita permite a una hormiga artificial volver por el camino que le ha llevado hasta la fuente de comida. Cuando una hormiga alcanza su destino cambia de comportamiento de *hacia adelante* a *hacia atrás* y comienza a construir el camino de retorno. Antes de comenzar el regreso a la fuente, la hormiga elimina los ciclos que pueda haber en el camino que ha construido mientras buscaba el nodo objetivo. El problema de los ciclos es que, mientras la hormiga realiza su viaje de retorno, puede recibir varias veces las aportaciones de feromona, generando un fenómeno de auto-reforzamiento de los ciclos.

Mientras la hormiga regresa, deposita una cantidad fija de feromona en los arcos que ha visitado. En particular, si una hormiga k en modo *hacia atrás* atraviesa el arco (i, j) cambia la cantidad de feromona del arco:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau^k \tag{12.2}$$

Con esta regla una hormiga que utilice el arco que conecta i y j incrementa la probabilidad de que lo usen el resto de hormigas.

La cantidad de feromona que se deposita está en función de la cantidad de hormigas que lo transitan.

En S-ACO la hormiga memoriza el camino que le ha llevado hasta la solución junto con el coste de los arcos que ha recorrido. De tal modo, que puede evaluar el coste de la solución obtenida y utilizarlo para ajustar la cantidad de feromona que deposita en cada arco que recorre, por medio de una función variable relativa al coste del camino, para que la búsqueda de caminos más cortos lleve más rápidamente hacia las mejores soluciones.

12.4.4 Evaporación de las Marcas de Feromona

La evaporación de feromona puede ser vista como un mecanismo que evita la rápida convergencia de las hormigas hacia una ruta que no es óptima. De hecho, la disminución de feromona que se encuentra en el camino favorece la exploración de nuevas rutas durante el proceso de búsqueda global.

[DCD98b] señala que en las hormigas reales este mecanismo también está presente, aunque no juega un papel fundamental. Supóngase que no es así, si no hubiera evaporación de feromona, las hormigas seguirían siempre el mismo camino. Gracias al proceso de evaporación las hormigas cambian periódicamente las zonas de exploración. Este es justamente el mecanismo que les permite sobrevivir y explorar zonas nuevas y tener disponibles otras rutas.

El mecanismo de evaporación de la feromona natural (y el de la artificial) juega un papel clave, porque sin él este sistema no funcionaría bien, como se ha podido ver en el tercer caso del experimento del doble puente. En las hormigas reales la intensidad de la feromona presente en el medio disminuye en función del tiempo. En S-ACO dicha evaporación es simulada aplicando una regla de reducción de feromona, que se muestra a continuación:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} \qquad \rho \in (0,1] \tag{12.3}$$

La evaporación de feromona hace que se construyan cada vez mejores soluciones, debido a que se evapora la feromona asociada a las primeras soluciones.

12.4.5 Meta-Heurística ACO

ACO constituye una meta-heurística. En otras palabras, es un método heurístico para resolver un tipo de problema computacional general. Suele aplicarse a problemas que no tienen un algoritmo que encuentre una solución satisfactoria o cuando no es posible implementar la solución óptima.

ACO resuelve problemas de optimización combinatoria, como encontrar la ruta más corta. El proceso distribuido de encontrar la ruta más corta es una fuente importante de investigación en inteligencia artificial. Los algoritmos ACO trabajan de una manera iterativa. En cada paso todas las hormigas artificiales contribuyen a dar una solución al problema utilizando la denominada matriz de feromona artificial. La matriz de feromona es actualizada con los valores asociados a las soluciones encontradas.

ACO se aplicó por primera vez al conocido problema del viajante ó *Traveling Salesman Problem* (TSP) [SG07]: un viajante partiendo de una ciudad tiene que visitar N ciudades sin repetirlas y volver al origen en el menor tiempo posible. En este problema se aplicó Ant System (AS) [DMC96]. En AS cada arista tiene asociado un valor de feromona artificial.

A la hora de abordar un problema computacional utilizando esta meta-heurística se identifican una seria de tareas o etapas:

1. Representar el problema como un conjunto de componentes y transiciones o como un grafo ponderado que recorren las hormigas para construir soluciones.
2. Definir de manera apropiada el significado de las marcas de feromona para la toma de decisión. Este es un paso crucial en la implementación de un algoritmo ACO y no es una tarea trivial.
3. Elegir la preferencia heurística de cada decisión que debe tomar una hormiga mientras construye una solución. Conviene reseñar que la información heurística es crucial para un buen rendimiento cuando se aplica a algoritmos de búsqueda local.
4. Implementar una búsqueda local eficiente.
5. Elegir un algoritmo ACO específico y apropiado.
6. Ajustar los parámetros del algoritmo ACO. Un buen punto de partida es utilizar configuraciones que han demostrado ser buenas en problemas similares. Otra posible alternativa es utilizar procedimientos automáticos de ajuste de parámetros.

Los pasos más importantes son los cuatro primeros, ya que una elección poco acertada en ellos hace que sea difícilmente subsanable mediante ajuste de parámetros.

12.5 Aproximación Paralela

ACO es una técnica fácilmente paralelizable por sus características distribuidas. Seguidamente se comentan los trabajos más representativos en la literatura.

[BKS98] constituye la primera aproximación paralela. Este método presenta limitaciones en su desarrollo cuando se analizan aspectos como el número de iteraciones locales, las reglas de asignación de tareas a los procesadores, las aproximaciones estáticas/dinámicas, etc.

[MM98] introduce una nueva técnica de paralelización ACO para resolver el problema de *Shortest Common Supersequence* (SCS), que tiene importantes aplicaciones en planificación de sistemas de producción, en ingeniería mecánica y en biología molecular. Emplea el *modelo isla* con varias colonias de hormigas que están separadas y que intercambian información según el rastro que siguen, pero en vez de usar un grafo (representación típica en ACO) para representar el problema, utiliza una cadena de caracteres, asignando un valor de feromona a cada carácter de la misma. Los resultados muestran que este algoritmo tiene una mejor heurística que un algoritmo genético, pero tiene la desventaja de que se pierde la funcionalidad que proporciona el uso de los grafos.

[Stü98] aplica una aproximación *maestro/esclavo* para paralelizar las diferentes técnicas de búsqueda de soluciones ACO con la característica de que éstas no interactúan. Stützle emplea una estrategia simple para ejecutar las sesiones independientes y paralelas de un algoritmo ACO. Las pruebas empíricas realizadas con el algoritmo MAX-MIN *Ant System* (AS) [DS04] al Problema del Viajante (TSP) demuestran la eficiencia de esta aproximación. Sin embargo, presenta el inconveniente de que depende tanto del problema en sí como del hardware disponible.

[DKGG01] implementa un nuevo sistema de paralelización ACO para problemas de programación industrial, probándolo en un procesador de memoria compartida con OpenMP. Esta técnica mejora los resultados de la aproximación secuencial, incrementándose notablemente la diferencia conforme se aumenta el tiempo de ejecución del algoritmo.

[RL02] analiza diferentes estrategias de paralelización que aplica específicamente al Problema del Viajante. Estas estrategias sólo son una guía para la paralelización de la meta-heurística ACO, no pudiendo considerarse una aproximación formal y genérica. Los resultados muestran un *speedup* aceptable, observándose que en problemas complejos se alcanza una mejor eficiencia, pero presenta el inconveniente de que requiere una gran cantidad de información, no siendo escalable.

Finalmente, [TG09] realiza una prueba para evaluar el rendimiento de la comunicación *Message Passing Interface* (MPI) multihilo. En esta aproximación se usan modelos de programación híbrida, combinando MPI a través de los nodos y *multithreading* dentro de un nodo, porque muchas implementaciones MPI están empezando a soportar comunicación MPI multihilo. Con esta técnica se consiguen mejores resultados al interrelacionarse los nodos de manera más eficiente.

12.6 Resumen

El objetivo de este capítulo ha sido introducir el algoritmo de optimización de la colonia de hormigas, abreviadamente conocido como meta-heurística ACO, que resulta particularmente apropiado para resolver problemas difíciles de optimización combinatoria. Este método meta-heurístico consiste en un conjunto de agentes (hormigas) artificiales que cooperan entre sí por medio de un conjunto de reglas que determinan la generación de información local y global y su actualización con el propósito de encontrar las mejores

soluciones. Se ha comenzado analizando el denominado experimento del doble puente para señalar las principales similitudes y diferencias entre las hormigas naturales y las hormigas artificiales. Posteriormente, se ha descrito el algoritmo S-ACO, que tiene su inspiración en el citado experimento y que resuelve el problema de encontrar la ruta más corta en un grafo dado. A continuación, se ha comentado la forma de abordar un problema computacional mediante esta meta-heurística. Finalmente, se han expuesto diversas técnicas de paralelización que aprovechan la naturaleza distribuida de este algoritmo.

Capítulo 13

Encaminamiento Adaptativo

El objetivo general de este capítulo es revisar los trabajos más representativos sobre protocolos de encaminamiento adaptativo para redes móviles ad hoc existentes en la literatura. En primer lugar se introduce la noción de encaminamiento adaptativo. Luego se presenta el encaminamiento ACO como un tipo particular de encaminamiento adaptativo. Seguidamente, se señalan las peculiaridades del encaminamiento ACO en el caso de su aplicación a redes móviles ad hoc. A continuación, se describen los principales protocolos de encaminamiento ACO para redes móviles ad hoc, haciendo énfasis en AntHocNet, el protocolo de referencia en el área. El capítulo finaliza con una breve síntesis de lo expuesto en el mismo.

13.1 Introducción

Se denomina encaminamiento adaptativo en una red al conjunto de técnicas o protocolos de encaminamiento que, como su nombre indica, intentan *adaptarse* a la variabilidad de la misma (tráfico, topología, etc.).

Dentro de éste, merece mención especial el denominado encaminamiento ACO ó conjunto de protocolos de encaminamiento que hacen uso de las técnicas ACO.

La aproximación ACO para el encaminamiento es bastante robusta porque la pérdida de hormiga/s no es importante. Esta aproximación es diferente de la aproximación de vector distancia, donde la información de encaminamiento procede de la información proporcionada por los nodos vecinos y de la aproximación de estado de enlace, donde la información de encaminamiento se actualiza con los mensajes recibidos de los nodos de la red.

13.2 Encaminamiento ACO

Un aspecto esencial del encaminamiento ACO es que las hormigas siempre muestran diversas rutas completas entre el origen y el destino, aumentando la sobrecarga respecto a una aproximación puramente reactiva.

Otra característica es la manera en la que las hormigas eligen una ruta. Ellas construyen el camino salto a salto de manera probabilística usando la información de feromona. El uso de ésta permite construir sobre la experiencia adquirida previamente por las hormigas. Esto es la clave de un proceso altamente distribuido. El hecho de que las hormigas construyan sus caminos de una manera probabilística permite la exploración de rutas múltiples. Esto hace que el algoritmo se adapte a los cambios de la red, incrementando tanto la robustez (a través de la disponibilidad de rutas de reserva) como el *throughput* de la red.

Una tercera característica es el reenvío estocástico de los paquetes de datos basado en la información de feromona, lo que asegura su encaminamiento por las mejores rutas. Si la feromona se mantiene actualizada por el uso de suficientes hormigas, el balanceo de carga sigue automáticamente los cambios en la red.

Todo lo anterior hace que los algoritmos de encaminamiento ACO presenten unas propiedades muy interesantes:

- Trabajan de una manera completamente distribuida: la información no se encuentra en un nodo central, sino que está contenida en cada nodo.
- Tienen gran adaptabilidad a los cambios de la red y del tráfico.
- Utilizan agentes móviles (hormigas) para determinar las rutas para el envío de los datos. Estos agentes son paquetes de control que se envían por la red. Hay dos tipos: hormigas *hacia delante* (van de origen a destino) y hormigas *hacia atrás* (van en sentido contrario).
- Pueden proporcionar encaminamiento multicamino.
- Presentan una excelente tolerancia a fallos, esto es, ofrecen un buen comportamiento ante el fallo de los agentes.
- Eligen automáticamente la ruta para el envío de los datos.

13.3 Encaminamiento ACO en Redes Móviles Ad Hoc

La aplicación directa del algoritmo ACO tal y como está descrito por Dorigo [Dor92, DS04] no es aconsejable en redes móviles ad hoc por la lenta convergencia que ofrece (cuando es el caso). Su propuesta se desarrolla en una topología estática de red, en la que se conocen de antemano todas las rutas. Lo único que realizan las hormigas es elegir la ruta en función de la carga de tráfico. En las redes móviles ad hoc, donde la topología es dinámica, las rutas no siempre son válidas. Una topología dinámica implica que se desconocen las rutas para realizar la comunicación. El primer paso es, por tanto, hacer una exploración [DDCG10] que capture rápidamente y sin mucho coste la topología de la red. [Gor00] señala que la exploración realizada por las hormigas recolectoras es *dirigida*: unas determinadas hormigas, que forman parte de lo que se denomina la *patrulla*, salen en diversas direcciones a explorar los alrededores de la colonia. Estas hormigas, al volver a la colonia, indican si han encontrado o no el alimento. En caso afirmativo, estimulan de alguna manera a las hormigas recolectoras a salir hacia la comida en la dirección indicada por ellas, creando un flujo de tráfico entre el hormiguero y la comida. Posteriormente, las hormigas recolectoras obtienen la ruta mínima, pero sólo en la zona indicada por la patrulla. El problema del cálculo de la ruta mínima es, pues, posterior a la exploración. Análogamente funciona ACO. Mediante un mecanismo similar de patrulla se sitúa al objetivo en una zona local. Posteriormente, se realizan búsquedas en profundidad en dicha zona.

13.4 Trabajos Relacionados

AntNet: Distributed Stigmergetic for Communications Networks [DCD98a] es el primer algoritmo ACO de encaminamiento para redes, en este caso para redes cableadas o estáticas (que no dinámicas). Es multicamino y se adapta al tráfico de la red, no siendo necesario

un exhaustivo cálculo de rutas. Tampoco requiere un mantenimiento de las mismas y no necesita actualizar una información global porque en el caso de que una ruta sufra modificaciones es el mismo nodo quien realiza esta tarea.

AntNet-FA (*Flying Ants*) [DCD98b] es una versión mejorada y escalable de AntNet, en el que las hormigas hacia delante hacen uso de las colas de alta prioridad al igual que las hormigas hacia atrás. Éstas últimas actualizan las tablas de encaminamiento en cada nodo visitado usando estimaciones locales de su tiempo de recorrido, y no del experimentado por las hormigas hacia delante. El rendimiento de AntNet-FA mejora con el tamaño de la red y su eficiencia es similar o incluso mejor que la de AntNet.

AntNet y AntNet-FA no son protocolos de encaminamiento ACO para redes móviles ad hoc pero son, sin duda, sus precursores.

Seguidamente se describen los protocolos de encaminamiento ACO más importantes, agrupándose éstos en proactivos, reactivos e híbridos.

Los *protocolos de encaminamiento ACO proactivos* más representativos son los siguientes:

Adaptive Swarm-based Distributed Routing [KESMI+02], más conocido como Adaptive-SDR, se inspira en AntNet. Este protocolo tiene la propiedad de agrupar los nodos en colonias para solucionar los problemas de escalabilidad de otros protocolos, derivados del hecho de cada nodo tenga que enviar una hormiga a todos los demás. La agrupación de nodos en colonias se realiza por una entidad central de control que se percata de sus posiciones geográficas. Existen dos tipos de hormigas: hormigas de colonia y hormigas locales. Las primeras tienen la misión de encontrar rutas desde una colonia a otra. Las hormigas locales son interiores a la colonia y encuentra rutas dentro de la colonia, apoyándose en dos tablas de encaminamiento. Este protocolo presenta numerosos inconvenientes: muchas colonias es desaconsejable por la sobrecarga que ocasionan, conocer el número óptimo de nodos que debe haber en una colonia no es nada trivial, en sistemas distribuidos no siempre se dispone de una entidad central de control como la que presupone, el gran procesamiento de las tablas de encaminamiento realizado por las hormigas locales implica un elevado consumo de recursos y requiere de dispositivos con importantes prestaciones, etc.

Mobile Ant-Based Routing [HB03], más conocido como MABR propone un esquema para abordar el problema de la escalabilidad en el encaminamiento en redes móviles ad hoc. Esta aproximación abstrae la topología dinámica de la red para obtener *encaminadores lógicos* y *enlaces lógicos*. Estos dos conceptos se refieren al conjunto de nodos y caminos creados entre ellos, respectivamente. Este algoritmo usa la partición geográfica del área del nodo y el direccionamiento geográfico de exploración de feromona. En este protocolo las hormigas hacia delante chequean periódicamente si un camino a un destino elegido aleatoriamente es funcional y las hormigas hacia atrás reflejan el estado actual de la red, haciendo que los caminos seguidos por las hormigas se refuercen positiva o negativamente. Además esta aproximación usa la evaporación de feromona, que favorece más la exploración y elimina los caminos no actualizados. El problema de esta propuesta es que se limita a presentar el modelo teórico, no aportando resultados experimentales.

Probabilistic Emergent Routing Algorithm for mobile ad hoc networks (PERA) [BM03] es un protocolo que ajusta en cada nodo la probabilidad de que cada uno de sus vecinos pueda recibir y reenviar el paquete de datos. Cada hormiga *hacia delante* contiene las direcciones IP del nodo origen y destino, un número de secuencia, un campo contador de saltos y una pila que crece dinámicamente. La pila contiene información de los nodos que la hormiga *hacia delante* visita así como los tiempos asociados. Cuando un nodo no tiene un registro de una ruta a un destino se crea una hormiga *hacia delante*, donde el nodo pone su propia dirección IP en la pila de dicha hormiga, así como el tiempo en el que

se ha creado la hormiga. A partir de este momento el nodo almacena periódicamente las hormigas *hacia delante* enviadas a los destinos para cuando la ruta sea requerida. Cuando esta hormiga *hacia delante* alcanza el destino, el nodo destino crea una hormiga *hacia atrás*. Este nuevo agente usa la información contenida en la hormiga *hacia delante* en el camino inverso para actualizar la distribución de probabilidad en cada nodo y reflejar el estado actual de la red. El hecho de que las hormigas *hacia delante* se envíen en modo *broadcast* desde el origen y en los nodos intermedios ocasiona un *broadcast* múltiple al encontrar diferentes caminos al destino, originándose una gran sobrecarga.

Ant Routing Algorithm for Mobile Ad hoc networks (ARAMA) [HS03] es un algoritmo de encaminamiento proactivo en el que las hormigas *hacia delante* no sólo tienen en cuenta el factor de contador de saltos (como la mayoría de los protocolos), sino que también valoran la heurística de enlace local a través de la ruta (como puede ser la energía de la batería del nodo y el retardo de la cola). El algoritmo define un valor llamado *grado*. Este valor es calculado por cada hormiga *hacia atrás* en función de la información del camino almacenado en la hormiga hacia delante. En cada nodo la hormiga *hacia atrás* actualiza la cantidad de feromona de la tabla de rutas del nodo usando el *grado*. El protocolo usa el mismo grado para actualizar el valor de feromona de todos los enlaces. Los autores afirman que la sobrecarga del descubrimiento y mantenimiento de rutas se reduce por medio del control del número de hormigas *hacia delante*. Sin embargo, no aclaran cómo controlar la generación de hormigas en un entorno dinámico.

AntNet Ring Search and Local Retransmission (AntNet-RSLR) [RMH11] es una adaptación de AntNet a redes móviles ad hoc mediante la incorporación de dos técnicas: *Expanding Ring Search* (ERS) y *Local Retransmission* (LR). En este protocolo los agentes móviles construyen caminos entre pares de nodos, explorando la red concurrentemente e intercambiando la información obtenida para actualizar las tablas de encaminamiento, lo que permite disminuir tanto la sobrecarga como el retardo extremo a extremo respecto a AntNet, AODV y DSR. Mediante la técnica de búsqueda del anillo expandido el mensaje de solicitud de descubrimiento de ruta se difunde progresivamente por inundación desde el nodo origen. Inicialmente se difunde el mensaje a una pequeña vecindad con un pequeño valor *Time To Live* (TTL), que se va incrementando hasta llegar al destino. Este mensaje es reenviado por el nodo origen si no recibe respuesta en un intervalo de tiempo. Si en la solicitud de ruta el valor TTL ha alcanzado un cierto valor umbral sin recibir respuesta asume que el destino es inalcanzable. Sin embargo, esto produce una gran sobrecarga y puede originar bucles que reducen el ratio de entrega de paquetes. Para solucionar el problema de la sobrecarga introduce una variante de esta técnica denominada *Blocking-ERS*, que no asume el procedimiento de búsqueda de una ruta desde el nodo origen cuando se requiere un nuevo envío en modo *broadcast*, generando un *rebroadcast* desde un nodo intermedio convenientemente elegido. La técnica de retransmisión local se usa cuando un nodo intermedio no recibe el correspondiente paquete de datos al expirar el valor del temporizador, enviando un mensaje de control de notificación negativa (*NACK*) para que el nodo intermedio (y no el nodo origen) vuelva a retransmitir el paquete de datos que falló. Esto tiene el inconveniente de que no se conoce a priori la capacidad del buffer del nodo que almacena los datos para su posible retransmisión. Al estar basado en un protocolo proactivo como es AntNet, la sobrecarga debería estar presente como aspecto negativo, si bien los autores afirman que se reduce.

Los *protocolos de encaminamiento ACO reactivos* más representativos son los siguientes:

Ant-colony based Routing Algorithm for mobile ad hoc networks (ARA) [GSB02] es protocolo reactivo en el que las entradas de la tabla de encaminamiento contienen valores

de feromona que facilitan la elección de vecino. Para conseguir un destino es preciso elegir un vecino que nos sirva de enlace y así sucesivamente hasta llegar al destino. En la tabla de encaminamiento los valores de feromona se degradan con el tiempo y los nodos entran en modo *sleep* si alcanzan un determinado umbral. El descubrimiento de ruta se realiza por inundación, esto es, las hormigas *hacia delante* son reenviadas a sus vecinos. Cada nodo que recibe esta hormiga actualiza su tabla de encaminamiento. Las hormigas *hacia adelante* duplicadas se identifican a través de un número de secuencia único y se eliminan. El nodo destino al recibir una hormiga *hacia delante* extrae su información y crea una hormiga *hacia atrás*, que regresa al nodo origen. Estas tienen una tarea similar a las hormigas *hacia delante*. Conviene señalar que la inundación tiene mayor alcance que el *broadcast* ya que los paquetes por inundación se transmiten a todos los nodos de la red por medio de multisalto, mientras que los paquetes por *broadcast* se transmiten sólo a los vecinos que están a un salto. El problema de la inundación es que conlleva una alta sobrecarga. Una vez que se ha realizado el descubrimiento de ruta para un destino determinado, el nodo emisor ya no genera un nuevo agente móvil hacia el destino, sino que el mantenimiento de ruta es realizado por los paquetes de datos. Los autores afirman que, en los escenarios considerados, el rendimiento de este protocolo es muy parecido al del DSR presentando una menor sobrecarga. No obstante, no incluyen escenarios que representen una carga de red elevada ni datos multimedia.

Ant-based Distributed Routing Algorithm for ad-hoc networks (ADRA) [ZGL04] es un algoritmo reactivo en el que las hormigas se mueven a través de la red entre pares de nodos elegidos aleatoriamente. Estas hormigas al moverse depositan feromona en función de varios parámetros: distancia en saltos desde su nodo origen, la calidad del enlace, la congestión encontrada en su viaje, la feromona actual que el nodo posee y la velocidad con que se mueven los nodos. Por supuesto, el mismo nodo, por la evaporación de feromona, cambia su valor de acuerdo con la edad del enlace. Una hormiga selecciona su camino en cada nodo intermedio según la feromona que tiene distribuida y para acelerar la elección del camino se dan parámetros con diferentes valores que actualizan la probabilidad en la tabla de encaminamiento. Los autores afirman que ADRA presenta un menor retardo medio extremo a extremo, una menor sobrecarga y un mejor ratio de entrega de paquetes que DSR. Asimismo, permite optimizar varios parámetros de QoS, tales como calidad de los enlaces, carga de los nodos, etc.

Ant Colony Based QoS Aware Routing Algorithm for MANETs [LF05] es un algoritmo de encaminamiento reactivo multicamino de enlace disjunto. La mayoría de los protocolos son esencialmente métodos de encaminamiento de ruta única, que tienden a tener una sobrecarga en los nodos que se encuentran en el camino más corto del origen al destino. Esta sobrecarga es debida a que en los métodos de camino único no existe balanceo de carga. El encaminamiento multicamino de enlace disjunto es más robusto, soportando mejor la QoS. Establece y utiliza múltiples rutas de enlace disjunto para enviar paquetes de datos y adaptar la feromona para dispersar el tráfico de la comunicación.

Una tendencia que ha cobrado fuerza en los últimos años es el diseño de protocolos especiales de encaminamiento único. Este tipo de protocolos tiene el inconveniente de que, al haber un único camino, si se produce una ruptura de enlace no se dispone de otra alternativa, siendo necesario realizar de nuevo un proceso de descubrimiento de ruta con el consiguiente aumento del retardo, con la sobrecarga de mensajes de control y con la disminución del ratio de entrega de paquetes. Para mejorar estos problemas surge *Efficient Ant-based Routing Algorithm for MANETs* [WSJX07]. Este protocolo ha sido diseñado para permitir que haya más caminos en los paquetes de petición/respuesta de rutas y descubrirlos con una menor sobrecarga.

Position Based Ant Colony Routing Algorithm for Mobile Ad Hoc Networks [KO08] es un algoritmo de encaminamiento reactivo basado en la localización de los nodos. El algoritmo encuentra las rutas óptimas o más cercanas en una red que contiene nodos de diferentes rangos de transmisión. Cada nodo asume su posición, la de sus vecinos y la del nodo destino. Sólo calcula la ruta en el momento de enviar datos desde un nodo origen a un nodo destino y éstos se envían cuando la ruta se establece. Para minimizar el tiempo que el algoritmo necesita para encontrar una ruta, la información sobre la posición de los nodos se usa como un valor heurístico. El uso de la información de localización como parámetro heurístico reduce significativamente el tiempo necesitado para establecer rutas desde el origen al destino así como el número de mensajes de control generados. Tiene un índice de entrega alto y un retardo medio de paquetes bajo, comparado con otros algoritmos de encaminamiento basados en posiciones. El algoritmo se estabiliza antes que AntNet.

Ant Routing Algorithms in Ad Hoc Networks with Critical Connectivity [RBR08] es un algoritmo que crea las rutas bajo demanda con el fin de disminuir la sobrecarga de encaminamiento con respecto a las aproximaciones proactivas. Las hormigas *hacia delante* sólo recopilan información de los nodos que recorren, eligiendo el siguiente salto hacia el destino basándose únicamente en la cantidad de feromona. La cantidad de feromona depositada por las hormigas *hacia atrás* sobre cada enlace recorrido es constante. La simplicidad del protocolo es útil para hacer un encaminamiento transparente en redes constituidas por elementos heterogéneos. En el algoritmo las tablas de encaminamiento en cada nodo son probabilísticas: el siguiente salto se selecciona de acuerdo con el mayor porcentaje de feromona dejado por las hormigas. Así, el reenvío de las hormigas *hacia delante* es probabilístico y permite la exploración de otras rutas disponibles en la red. Los paquetes de datos son enviados de forma *determinada* (*unicast*) por los nodos intermedios que se encuentran en el camino desde el emisor al nodo destino. Este proceso crea una ruta global mediante el uso de información local.

El protocolo *imProved Ant Colony Optimization routing algorithm for mobile ad hoc NETworks* (PACONET) [OTT08] es un protocolo de encaminamiento reactivo donde las hormigas *hacia delante* exploran los caminos de la red en busca de rutas desde un origen a un destino en modo *broadcast* restrictivo y las hormigas *hacia atrás* establecen la información del camino adquirido por las hormigas *hacia delante*. Los paquetes de datos son enviados probabilísticamente hacia los nodos que tienen la mayor concentración de feromona. Las hormigas *hacia delante* viajan hacia los nodos no visitados, pero si no los encuentran siguen la ruta de los nodos con mayor concentración de feromona. Las filas de la tabla de rutas representan a los vecinos de un nodo y las columnas representan a los nodos de la red. Cada par (fila, columna) en la tabla de encaminamiento tiene dos valores: (a) un valor binario que indica si el nodo ha sido visitado, y (b) la concentración de feromona. Las hormigas hacia delante sólo tienen en cuenta la concentración de feromona después de que todos los vecinos de una columna hayan sido visitados. El propósito de esto es asegurar que todos los caminos sean explorados para encontrar el mejor camino hacia el destino. El nodo con la mayor feromona es elegido como el siguiente salto, después de que la hormiga *hacia adelante* haya determinado que el nodo no ha sido visitado antes.

Los *protocolos de encaminamiento ACO híbridos* más representativos son los siguientes:

Mobile Agents based Routing Protocol for Mobile Ad Hoc Networks [MTS02], más conocido como Ant-AODV, es una forma híbrida de encaminamiento basado en ACO y en el protocolo de encaminamiento AODV. Para superar algunos de los inconvenientes de AODV, como es la sobrecarga generada por el aumento de mensajes de control, se emplea esta técnica híbrida que destaca la conectividad de los nodos y disminuye el retardo extremo a extremo, así como la latencia del descubrimiento de ruta. Las hormigas Ant-AODV

trabajan independientemente y proporcionan rutas a los nodos. Los nodos también tienen la capacidad de realizar un descubrimiento de ruta bajo demanda para los destinos que no tienen una entrada de ruta lo suficientemente actualizada. El uso de hormigas con AODV incrementa la conectividad de los nodos, que está asociada al número de destinos que tiene dicho nodo y a los que se llega por medio de la correspondiente entrada actualizada en la tabla de encaminamiento. Ante la petición de ruta RREQ, la probabilidad de recibir una respuesta más rápida es mayor al haber más nodos conectados. Como las hormigas actualizan las rutas continuamente, un nodo emisor puede seleccionar una nueva ruta y más corta. Esto conduce a una disminución considerable en el retardo medio extremo a extremo, comparado con AODV y con el encaminamiento basado en ACO. Además Ant-AODV usa mensajes de error de ruta (RERR) para informar en cadena a otros nodos de un fallo de enlace local de forma similar a como hace AODV.

AntHocNet [DC04, DCDG04] es un algoritmo de encaminamiento ACO híbrido, multicamino y adaptativo. Data de 2004 y en estos casi diez años ha tenido numerosas extensiones y variaciones para mejorar su rendimiento. Como se ha comentado anteriormente, AntHocNet es un algoritmo híbrido (reactivo y proactivo), multicamino y adaptativo. Es reactivo porque tiene agentes que actúan bajo demanda para establecer los caminos hacia los destinos. Es proactivo porque posee agentes que actúan con la información recogida y tienen la posibilidad de descubrir nuevas rutas para tener alternativas ante fallos de enlace. Es multicamino porque establece diferentes caminos para enviar la información al destino. Finalmente, es adaptativo porque se adecua a las condiciones del tráfico y de la red.

AntHocNet sigue una estructura parecida a AntNet-FA, pero difiere en sus características. Como se ha visto anteriormente, AntNet-FA se aplica a topologías de redes estáticas, en las que se conocen las rutas y la convergencia es lenta. Por tanto, lo único que tienen que hacer las hormigas es elegir el camino. AntHocNet, por su parte, tiene en cuenta la topología dinámica y demás características de las redes ad hoc. Cuando cambia la topología de la red es necesario restablecerla rápidamente y se hace por medio de un nuevo proceso de descubrimiento de ruta. Si se destinan muchos recursos para acelerar este proceso, aumenta el intercambio de información, lo que puede llegar a colapsar la red. Hay, por tanto, un problema: si no se quiere sobrecargar la red, se aumenta el tiempo de convergencia del algoritmo ACO; si se quiere disminuir el tiempo de convergencia, se sobrecarga la red. En otras palabras, AntHocNet es una modificación del algoritmo AntNet-FA que acelera su tiempo de convergencia sin sobrecargar la red. El principal desafío de las MANETs, cuando se aplica a esquemas de encaminamiento basado en las hormigas, consiste en encontrar el correcto equilibrio entre las tasas de generación de agentes y la sobrecarga asociada. La propiedad adaptativa, que se da en AntHocNet, lo adecua a las condiciones del tráfico y de la red.

En el comportamiento de AntHocNet se distinguen varias fases:

i) Establecimiento de la información de encaminamiento: se inicia con el envío, bajo demanda, de agentes para el cálculo de la ruta al destino. Esta fase se apoya en la propiedad multicamino, que se considera de gran relevancia, debido a que se tienen que crear las rutas lo antes posible, de tal forma que se pierda el menor número de paquetes de datos.

ii) Encaminamiento de los datos: En esta fase los datos se envían de forma estocástica y *unicast* valiéndose de las feromonas de las tablas de encaminamiento que tienen localmente todos los nodos. Esta estrategia de encaminamiento pretende expandir la carga de los datos consiguiendo un mejor balanceo de la carga.

iii) Mantenimiento de las rutas establecidas y exploración de otras nuevas: en esta fase, mientras una sesión de datos está preparada para retransmitir la información, se envían hormigas proactivas de mantenimiento según la tasa de envío de datos. La finalidad de esta fase es actualizar la calidad de los enlaces de las rutas y los valores de feromona entre el camino que va desde el origen al destino.

iv) Gestión de los fallos de enlace: En esta fase los nodos pueden detectar fallos de enlace. Una vez que se detectan, AntHocNet los intenta mitigar valiéndose de diferentes mecanismos como el envío de mensajes de notificación de fallo y la reparación local de ruta.

Las simulaciones analizan diferentes escenarios en término de número de nodos, movilidad y densidad de los nodos. En general, en todos ellos AntHocNet tiene un comportamiento similar o incluso mejor que AODV. En particular, se obtuvo mejor ratio de entrega de paquetes que AODV, un retardo extremo a extremo ligeramente mayor en AntHocNet que en AODV para los escenarios más simples (alta densidad y caminos cortos), mejorando en AntHocNet para los escenarios más complejos. Finalmente, AntHocNet tiene también un buen rendimiento para grandes redes, por lo que constituye un protocolo escalable.

A unicast routing protocol for mobile ad hoc Networks using Swarm Intelligence (ANSI) [RS06] presenta un protocolo en el que la tabla de encaminamiento contiene una entrada por cada nodo que se alcanza y el siguiente mejor salto, mientras que las tablas de decisión de la hormiga almacenan los valores de feromona. En este protocolo las hormigas *hacia delante* sólo se generan cuando un nodo necesita transmitir datos a otro nodo. Estas hormigas se envían en modo *broadcast*, mientras que las *hacia atrás* lo hacen en modo *unicast*, siguiendo la pista que han dejado las hormigas *hacia delante* en el camino y actualizando los valores de feromona de los nodos. Los paquetes de datos eligen el siguiente salto teniendo en cuenta el mayor valor de feromona. El protocolo es tan bueno o incluso mejor que AODV con respecto a la entrega de paquetes y al retardo extremo a extremo.

La Tesis de Ducatelle [Duc07] es una evolución de AntHocNet, respecto a la Tesis de Di Caro [DC04] y otros trabajos como [DCDG04]. Las diferencias entre ambas versiones están en el empleo de distintos mecanismos en el proceso de establecimiento reactivo de ruta y en el proceso proactivo de mantenimiento de ruta.

En lo referente al proceso de descubrimiento, las versiones más antiguas crean múltiples rutas en este proceso. Con esta estrategia se tiene la ventaja de que múltiples rutas están disponibles desde el comienzo de la sesión de datos, de forma que la sesión está mejor protegida ante los fallos de enlace y puede comenzar inmediatamente el balanceo de la carga. Sin embargo, esto puede conducir a una alta sobrecarga que a veces es innecesaria. Esto último se experimentó decidiéndose entonces crear sólo una ruta en el proceso de descubrimiento y obtener múltiples caminos en el proceso de mantenimiento de rutas.

Con respecto al proceso de mantenimiento de ruta, las versiones predecesoras consisten en envíos de hormigas proactivas para la exploración de rutas, no aplicando el proceso de difusión de feromona. Estas hormigas proactivas se envían, por tanto, en modo *unicast* y también en modo *broadcast* con una cierta probabilidad, porque hay que explorar nuevas rutas sin el uso de la difusión de feromona. Si bien se reduce la sobrecarga por no utilizar difusión de feromona (los mensajes *Hello* son más sencillos ocupando menos memoria en bytes), el envío en modo *broadcast* provoca una exploración completamente ciega porque las hormigas proactivas se envían a muchos nodos innecesariamente. Tras el análisis de estas dos aproximaciones se comprueba finalmente que el proceso de mantenimiento proporcionado por la versión de Ducatelle es más efectiva y eficiente que sus predecesoras.

[WDR08] introduce varias modificaciones a AntHocNet en la fase de establecimiento de

ruta para controlar el número de hormigas que se mueven por la red y actualizar convenientemente los valores de feromona en todos los nodos intermedios cuando se ha establecido la ruta. Las simulaciones demuestran que reduce la sobrecarga y el retardo extremo a extremo, mientras que el ratio de entrega de paquetes se mantiene igual comparado con AntHocNet.

[KD08] incluye en AntHocNet la propiedad de multicamino de nodos disjuntos. El protocolo facilita especialmente el balanceo de carga y, en menor medida, la tolerancia de fallos y la reducción del retardo extremo a extremo. Las rutas múltiples generadas por AntHocNet no son disjuntas. Estas rutas pueden tener nodos y enlaces en común, presentando desventajas respecto a las rutas disjuntas que proporciona este algoritmo. Rutas disjuntas de enlaces o nodos son aquellas que para una misma sesión de datos no comparten nodos o enlaces, respectivamente. La existencia de estas rutas de nodos disjuntos permite que la carga se distribuya mejor.

[DDCG08] es una variante de AntHocNet cuya parte proactiva se ejecuta en segundo plano ofreciendo un servicio de encaminamiento *best effort* y cuya parte reactiva ofrece un servicio orientado a conexión. Su propiedad más importante es que permite elegir por separado entre el encaminamiento proactivo y reactivo para cada sesión de datos. Los paquetes de datos pueden seguir tanto rutas proactivas como reactivas. Existe una sinergia (cooperación interactiva) porque la parte reactiva del algoritmo confía en la información de encaminamiento proactiva. Las simulaciones demuestran que esta variante mejora los resultados obtenidos por sus predecesores.

[DCDG08] realiza el estudio del rendimiento de AntHocNet y AODV en un entorno urbano utilizando aplicaciones en tiempo real. Se realiza una simulación realista en términos de radio de propagación, restricción de movilidad de los nodos y tráfico de los datos. En otras palabras, el escenario es real con obstáculos y patrones de tráfico reales. Para las simulaciones se usa QualNet. En la mayoría de los escenarios AntHocNet mejora a AODV en términos de ratio de entrega, retardo y *jitter*. Además, en la mayoría de las pruebas, AntHocNet tiene también menor sobrecarga que AODV y OLSR. En escenarios urbanos, AntHocNet tiene la ventaja de que la densidad local (número de vecinos) experimentada por cada nodo es relativamente baja y crece lentamente. Se observa que la cantidad de nodos influye mucho en el ratio de entrega, mientras que el movimiento de los nodos no parece que tenga tanta influencia. También se observa que la densidad de nodos tiene un fuerte impacto en el ratio de entrega, mientras que la velocidad de los nodos parece tener relativamente una menor repercusión.

AntHocNet [DC04, DCDG04] y todas sus variantes [Duc07, WDR08, KO08, DCDG08] constituyen una referencia en el área de los protocolos de encaminamiento ACO para redes móviles ad hoc. Su carácter adaptativo le confiere unas propiedades especiales haciendo que sus métricas de funcionamiento, en términos generales, sean mejores que las de cualquier otro protocolo de encaminamiento para redes móviles ad hoc. No obstante lo anterior, en escenarios altamente dinámicos presenta problemas de escalabilidad.

HOPNET: Hybrid ant colony OPtimization routing algorithm for mobile ad hoc NETworks [WOTT09] es un algoritmo de encaminamiento híbrido basado en hormigas que saltan de una zona a otra. El algoritmo tiene características extraídas de los protocolos ZRP y DSR, siendo altamente escalable comparado con otros protocolos híbridos. Este algoritmo descubre proactivamente la ruta dentro de la zona de vecindad de un nodo, realizando la comunicación entre zonas de forma reactiva. El tamaño de la zona lo determina la longitud del radio medido en saltos y no el nodo. Por consiguiente, la zona de encaminamiento la constituyen los nodos que están dentro de la longitud de radio especificado. Un nodo puede pertenecer a múltiples zonas que se solapen y las zonas pueden variar de

tamaño. Los nodos pueden ser clasificados como interiores o fronterizos (o periféricos). Los nodos fronterizos son los que están a la distancia del radio, mientras que los que se encuentran a una distancia menor del radio son interiores. Cada nodo tiene dos tablas de encaminamiento: tabla de encaminamiento intrazona (*Intrazone Routing Table* (IntraRT) y tabla de encaminamiento interzona (*Interzone Routing Table* (InterRT)). La zona IntraRT es mantenida proactivamente por lo que un nodo puede obtener rápidamente un camino hacia cualquier otro nodo dentro de esta zona. Esto se realiza enviando periódicamente hormigas *hacia delante* que detectan los caminos dentro de la zona y el cambio en la topología (nodos que se salen de la zona, fallos de enlace, nuevos nodos que entran, etc.). Cuando una hormiga *hacia delante* alcanza un destino, se envía una hormiga de retorno (*hacia atrás*) a través del camino descubierto. La zona InterRT almacena el camino a un nodo más allá de su zona. Esta tabla de rutas se establece bajo demanda y los nodos periféricos son encargados de enlazar las zonas. Cuando el número de nodos es pequeño el movimiento continuo de los nodos periféricos hace que constantemente haya que descubrir nuevas rutas, lo que provoca más retardo que en otros protocolos de encaminamiento híbridos.

Hybrid Routing Algorithm Based on Ant Colony and ZHLS Routing Protocol for MANET [RAP10] es un protocolo que combina ideas de ACO con el protocolo Zone-based Hierarchical (ZHLS). Funciona de forma similar a HOPNET [WOTT09], comportándose de forma proactiva dentro de una zona y realizando la comunicación de manera reactiva entre las diferentes zonas. Los autores afirman que este protocolo presenta un mejor ratio de entrega de paquetes, una menor sobrecarga y un menor retardo extremo a extremo que los algoritmos tradicionales.

13.5 Resumen

El principal objetivo de este capítulo ha sido conocer el estado del arte de los protocolos de encaminamiento adaptativo para redes móviles ad hoc basados en el algoritmo de optimización de la colonia de hormigas. Esta revisión se ha iniciado con los protocolos AntNet y AntNet-FA, protocolos de encaminamiento ACO para redes cableadas o estáticas, que son los precursores de los diferentes protocolos de encaminamiento ACO para redes móviles ad hoc. Posteriormente, se han analizado en orden cronológico los más representativos de la literatura, dividiendo este recorrido en protocolos proactivos, reactivos e híbridos, haciendo énfasis en estos últimos y, particularmente, en AntHocNet, protocolo híbrido de encaminamiento multicamino que constituye el antecedente inmediato del trabajo desarrollado en esta memoria.

Capítulo 14

Protocolos de Encaminamiento Adaptativo para Redes Móviles Ad Hoc

Este capítulo presenta el diseño y especificación de una familia de protocolos de encaminamiento ACO para redes móviles ad hoc. Este conjunto parte de un protocolo base denominado protocolo de encaminamiento ACO optimizado o, más comúnmente, AntOR, como se cita en lo sucesivo, expresión que corresponde al acrónimo de su denominación inglesa Ant Optimized Routing. AntOR presenta dos variantes principales: disjunto de enlace (AntOR-DLR) y disjunto de nodo (AntOR-DNR). A su vez, cada una de estas son precursoras de diferentes protocolos. El capítulo comienza señalando los aspectos más importantes de AntOR, las fases de que consta y las principales diferencias respecto a AntHocNet, protocolo en el que se inspira AntOR. Posteriormente, describe las estructuras de datos utilizadas por este protocolo, estructuras que son básicamente comunes al conjunto de protocolos que se derivan de él. A continuación, se especifica el funcionamiento del mismo, analizando detalladamente en cada fase las novedades introducidas respecto a AntHocNet. Una vez descrito AntOR, el capítulo continúa con la especificación de los diferentes protocolos que derivan de aquel, empezando por la versión disjunta de enlace (AntOR-DLR) así como sus variantes AntOR-RDLR, AntOR-UDLR, AntOR-v2 y HACOR. Luego, se analiza la versión disjunta de nodo (AntOR-DNR) así como sus aproximaciones paralelas PAntOR y PAntOR-MI. El capítulo finaliza con una breve síntesis de lo expuesto en el mismo.

14.1 Ant Optimized Routing (AntOR): Generalidades

AntOR está inspirado en el algoritmo AntHocNet, más concretamente, en la versión especificada en la Tesis de Ducatelle [Duc07], heredando de éste sus características de protocolo híbrido (reactivo y proactivo), multicamino y adaptativo. Al igual que su predecesor, presenta las siguientes fases:

- Establecimiento de ruta: al comienzo de la sesión de datos el nodo fuente, bajo demanda, envía agentes para descubrir las rutas disponibles al destino.
- Encaminamiento de datos: los datos se envían a través de los nodos hasta el destino valiéndose de la información de las rutas, pudiendo utilizar la técnica multisalto, esto es, enviando datos a través de nodos intermedios que funcionan como encaminadores.

- Mantenimiento de rutas establecidas y exploración de nuevas rutas: se actualiza la información de las rutas existentes y se intentan descubrir otras nuevas. Esta fase consta de dos etapas: difusión de feromona y envío proactivo de hormigas.

- Gestión de fallos: estos acontecen porque un nodo se sale del alcance de la red o porque no recibe los mensajes de control que se encargan de informar a un nodo de sus vecinos más próximos.

Asimismo, y análogamente a su predecesor, con independencia de la fase en la que se encuentre, cada proceso *hacia adelante* (desde el nodo emisor al nodo destino) tiene su correspondiente proceso *hacia atrás*.

En cuanto a las principales diferencias de AntOR respecto a AntHocNet, éstas consisten básicamente en la introducción de los siguientes elementos/procesos:

- Especificación de rutas disjuntas de enlace o de nodo.

- Separación de las feromonas en el proceso de difusión.

- Utilización de la métrica *distancia* en la exploración de caminos.

Estas tres características influyen especialmente en las fases 1 y 3 del algoritmo, esto es, en la fase de establecimiento y en la de mantenimiento y exploración de nuevas rutas.

14.2 Ant Optimized Routing (AntOR): Estructuras de Datos

Al igual que la práctica totalidad de protocolos de encaminamiento ACO, AntOR precisa de dos estructuras de datos: la Tabla de Encaminamiento y la Tabla de Vecinos. Estas tienen una funcionalidad similar a la que poseen en otros protocolos de encaminamiento. Seguidamente se especifica cada una de ellas.

14.2.1 Tabla de Encaminamiento

Como en todo algoritmo ACO de encaminamiento para redes móviles ad hoc, la información relativa al encaminamiento se organiza en las denominadas tablas de encaminamiento. Esta estructura de datos está presente también en protocolos de encaminamiento ACO para redes cableadas como AntNet o en protocolos clásicos de encaminamiento para redes móviles ad hoc como AODV.

Estas tablas contienen la información utilizada por el algoritmo en sus decisiones locales de reenvío. La clase de información que contiene, así como la manera en que se usa y actualiza, depende exclusivamente de las características del algoritmo. La tabla de encaminamiento es a la vez una base de datos local y un modelo local del estado global de la red.

Esta tabla está formada por los siguientes campos:

- *Feromona regular* (τ_{ij}^{d}): Indica el camino por donde viajan los datos. Es un valor heurístico que contiene una estimación de la *bondad* (buena calidad) para retransmitir los paquetes de datos por la ruta que va desde i al destino d con siguiente salto j. Este valor se expresa como la inversa de una estimación de tiempo o *coste* como se explicó cuando se introdujo la ecuación 14.7. Este *coste* se basa en la métrica empleada para la evaluación del algoritmo.

- *Feromona virtual* (ω_{ij}^{d}): Indica un camino que posiblemente puede ser bueno. Este valor heurístico virtual tiene la misión de valor *auxiliar* y se utiliza como alternativa. Se crea o actualiza en el proceso de difusión de feromona.

- *Número medio de saltos* (h_{ij}^{d}): Se utiliza en el proceso de reparación local de rutas para indicar cuánto tiempo necesita el proceso para ejecutarse correctamente.

Los valores de *feromona regular* y *número medio de saltos* están relacionados de la siguiente forma: cuando una ruta tiene valor de uno también dispone del otro. Esto es debido a que estos dos valores están involucrados en el uso de las hormigas *hacia atrás* en el proceso reactivo, en el proactivo (exploración de nuevas rutas alternativas) y en el de reparación local de ruta. No obstante esto, el valor de *feromona virtual* se crea o actualiza de modo independiente, porque se utiliza en el proceso de difusión de feromona.

La Tabla 14.1 muestra la estructura de la tabla de encaminamiento en AntOR. Esta estructura almacena la siguiente información para cada entrada: destino alcanzable de la sesión de datos, siguiente salto por el cual se encaminan los datos, valor de *feromona regular* y de *feromona virtual*, y *número medio de saltos*.

Esta tabla crece dinámicamente conforme se vayan calculando rutas alcanzables.

Tabla 14.1: Tabla de encaminamiento (AntOR)

Valores / Entradas	**Destino**	**Siguiente Salto**	**Valor de Feromona Regular (τ)**	**Valor de Feromona Virtual (ω)**	**Número medio de saltos (h)**
$Entrada_1$	$Destino_1$	$Siguiente\ Salto_1$	τ_1	ω_1	h_1
$Entrada_2$	$Destino_2$	$Siguiente\ Salto_2$	τ_2	ω_2	h_2
...	...	...	...	...	...
$Entrada_i$	$Destino_i$	$Siguiente\ Salto_i$	τ_i	ω_i	h_i
...	...	...	...	...	...

La Figura 14.1 muestra que el nodo intermedio B tiene dos rutas asociadas a dos sesiones de datos con destinos G y D, respectivamente. La ruta con origen E y destino G (línea negra) y la ruta con origen A y destino D (línea roja) tienen en común el nodo intermedio B.

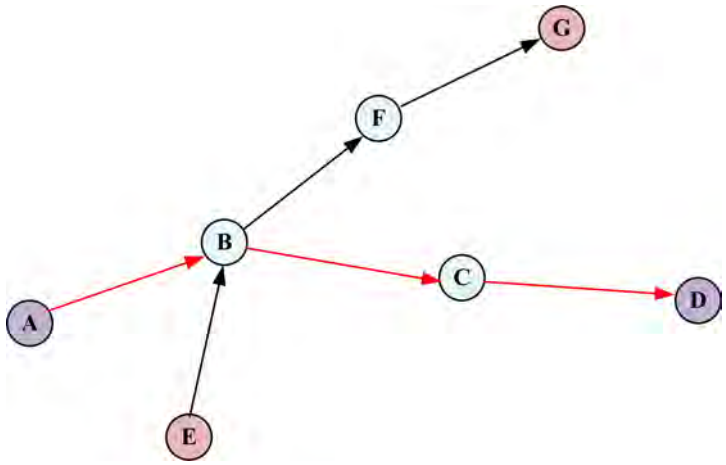


Figura 14.1: Escenario actualización de la tabla de encaminamiento (AntOR)

La Tabla 14.2 representa su tabla de encaminamiento. En esta tabla se aprecia cómo los dos procesos de establecimiento de ruta a los nodos G y D conllevan que haya tanto un valor de feromona regular τ como un valor h asociado al número medio de saltos. También se observa que no se dispone de un valor de feromona virtual ω porque en este caso no hay difusión de feromona.

Tabla 14.2: Estructura de encaminamiento del nodo B (AntOR)

Ruta del Nodo B	**Destino**	**Siguiente Salto**	τ	ω	**h**
Entrada 1	G	F	τ_1	-	h_1
Entrada 2	D	C	τ_2	-	h_1

14.2.2 Tabla de Vecinos

Esta estructura de datos contiene la información que cada nodo tiene de los vecinos a un salto con su correspondiente tiempo de *escucha*. La tabla de vecinos mantenida por el nodo i es un vector con una entrada por cada uno de sus vecinos. Cada entrada corresponde a la información que el nodo i tiene de la presencia del nodo vecino j así como un valor de *tiempo* que indica cuándo fue la última vez que lo escuchó, esto es, que i recibió un mensaje de j. Esta estructura se utiliza, como su nombre indica, para indicar la presencia de los vecinos y detectar posibles fallos de enlace.

La Tabla 14.3 representa la tabla genérica de vecinos de AntOR. En esta estructura cada nodo local tiene una lista de vecinos a 1 salto con la siguiente información: un identificador del vecino $Id\ Vec_k$ y el valor del último tiempo $Tiempo\ Vec_k$, asociado al mensaje de notificación de vecindad (*Hello*) que el nodo vecino le envió.

Tabla 14.3: Tabla de vecinos (AntOR)

Nodo local	Id Vec_1	Id Vec_2	...	Id Vec_k	...	Id Vec_N
	Tiempo Vec_1	Tiempo Vec_2		Tiempo Vec_k		Tiempo Vec_N

La Figura 14.2 muestra un ejemplo de una tabla de vecinos. Se puede apreciar que corresponde a una red con 3 nodos que intercambian mensajes entre sí para actualizar sus tablas de vecinos.

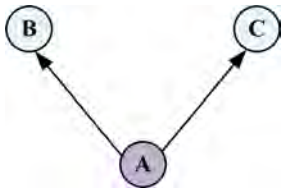


Figura 14.2: Escenario que representa la vecindad del nodo A (AntOR)

Tomando como punto de referencia el nodo A, después de recibir los correspondientes mensajes de B y C, su tabla de vecinos quedaría tal y como indica la Tabla 14.4. Esta tabla representa el identificador ID de cada uno de los nodos vecinos del nodo A, así como el tiempo de escucha actualizado.

Tabla 14.4: Estructura de vecinos del nodo A (AntOR)

Nodo A	Id Vecino B	Tiempo Vecino B
	Id Vecino C	Tiempo Vecino C

14.3 Ant Optimized Routing (AntOR): Funcionamiento

14.3.1 Establecimiento de Ruta

En un principio los nodos no disponen de información de encaminamiento para enviar los datos, pero sí tienen las aplicaciones para empezar: generador de tráfico, *ftp*, *ping*,..., los interfaces de red, la pila de protocolos (IP, UDP/TCP, etc.). La aplicación genera los datos en el nodo, pero al no tener ruta disponible no los puede enviar. El nodo necesita, por tanto, enviar unos agentes reactivos (hormigas reactivas) que descubran las rutas al destino.

14.3.1.1 Proceso Reactivo Hacia Adelante

Al comienzo del proceso de establecimiento de ruta, el nodo s, origen de la sesión de datos, crea una hormiga reactiva *hacia adelante*, más conocida por su abreviatura inglesa, *Reactive Forward Ant* (RFA). Esta hormiga es un paquete de control que tiene como objetivo encontrar un camino desde s a un destino dado d. Esta hormiga va desde el nodo fuente al nodo destino, siendo enviada por s en modo *broadcast*.

Los nodos intermedios que reciben esta hormiga la reenvían en el proceso de búsqueda de ruta hasta llegar al destino. Este tipo de hormigas dispone de una lista P de nodos visitados para que no se repitan los nodos intermedios.

El modo de reenvío de la RFA en los nodos intermedios puede ser *unicast* o *broadcast*, dependiendo de si el nodo actual tiene información disponible de encaminamiento al destino d. Por lo general, las RFAs se envían en modo *broadcast* porque se pretende descubrir la primera ruta. El modo *unicast* se utiliza siempre que el nodo actual tenga información de un encaminador vecino que sirva para retransmitir la correspondiente RFA al siguiente salto. En otras palabras, un nodo tiene información de encaminamiento siempre que se realiza el establecimiento de ruta, utilizando en el primer establecimiento el modo *broadcast* para el envío de las RFAs y en los posteriores (debido a fallos de enlace en los nodos origen) este modo o el *unicast*, debido a restos de rutas pertenecientes a otros establecimientos anteriores.

El reenvío *unicast* se realiza probabilísticamente utilizando la ecuación 14.1, donde τ_{in}^d es el valor de feromona regular del enlace que va del nodo i al siguiente salto n en la ruta hacia el destino d, N_i^d es el conjunto de vecinos del nodo i con una ruta disponible a d y β_1 es un parámetro de ajuste que influye en el comportamiento exploratorio de las hormigas.

$$P_{in}^d = \frac{(\tau_{in}^d)^{\beta_1}}{\sum_{j \in N_i^d} (\tau_{ij}^d)^{\beta_1}} \qquad \beta_1 \geq 1 \tag{14.1}$$

El valor de β_1 se determina experimentalmente. Si se utiliza un valor alto de β_1, las rutas con una mayor concentración de feromona regular son las candidatas para retransmitir las RFAs, obteniéndose rápidamente la ruta inicial. Si, por el contrario, se le asigna un valor inferior, las rutas tienden a ser elegidas con probabilidad similar.

Más detalladamente, el proceso de selección de ruta de la ecuación 14.1 es como sigue:

Cuando un nodo tiene la posibilidad de realizar el salto a sus nodos vecinos para llegar al destino d, calcula las probabilidades P_{in}^d de cada uno de estos vecinos n con el valor de feromona regular. De acuerdo con esta estrategia, no se eligen a priori las rutas que se van a utilizar, sino que se seleccionan como sigue:

- Se genera con un número aleatorio *rand* con probabilidad uniforme entre 0 y 1.
- Se calculan los intervalos no solapados asociados a las probabilidades P_{in}^d calculadas anteriormente. Estos intervalos se asocian a cada nodo vecino posible a la hora de seleccionar el candidato a transmitir el mensaje.
- Una vez obtenido *rand*, se elige la ruta asociada al intervalo que corresponde con P_{in}^d. Para ello se reenvían las hormigas reactivas al siguiente salto n que tiene como destino d.

La ecuación 14.1 está basada en un mecanismo de selección, ampliamente utilizada en los algoritmos genéticos, denominado *selección de ruleta*. Este mecanismo también se conoce como selección proporcional a la función de desempeño. Siendo N el número de individuos existentes y f_i el desempeño del i-ésimo individuo, la probabilidad asociada a su selección está dada por la siguiente ecuación:

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_i} \tag{14.2}$$

En la Figura 14.3 se muestra un ejemplo de la selección de un individuo. Se comprueba que la condición:

$$\sum j = N_i^d(\tau_{ij}^d)^{\beta_1} = F$$

es decir, que la suma de todos los valores de feromona regular elevado a un exponente es la *aptitud* total. En este ejemplo hay 4 intervalos etiquetados con letras, eligiéndose el B porque el número aleatorio *rand* está comprendido en él.

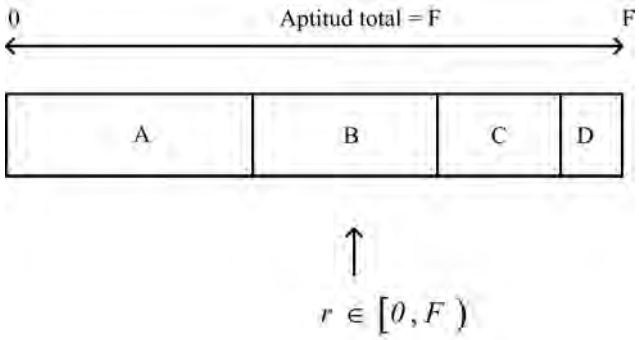


Figura 14.3: Ejemplo de selección de un individuo

Por otro lado, como se mencionó anteriormente, cuando el nodo actual no dispone de información de encaminamiento a un destino, se reenvía siempre la hormiga en modo *broadcast*. Debido a estos procesos de difusión (incluyendo el *broadcast* inicial en el nodo origen s), una hormiga reactiva *hacia adelante* puede proliferar rápidamente. Esto genera diferentes copias de la hormiga que siguen distintos caminos al destino. AntOR limita la sobrecarga que se origina, haciendo que los nodos sólo reenvíen la primera copia de la hormiga que reciben, descartando las posteriores.

La Figura 14.4 ilustra el primer proceso de establecimiento de ruta.

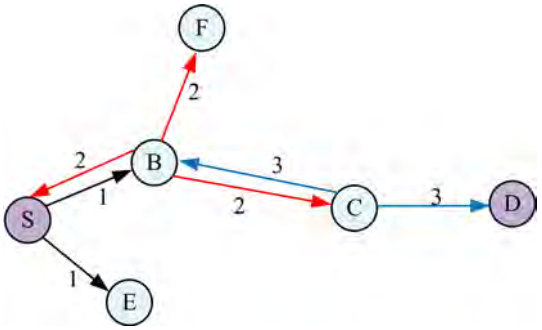


Figura 14.4: Proceso del primer establecimiento de ruta (AntOR)

Las flechas de la citada figura indican el rango de transmisión. El esquema es como sigue: el nodo S inicia el primer establecimiento de ruta hacia el nodo destino D enviando una RFA en modo *broadcast*. A continuación, los nodos B y E reciben la hormiga (proceso 1) para posteriormente reenviarla. El nodo B se encuentra con dos alternativas de envío: *unicast* o *broadcast*. Al estar en el primer establecimiento de ruta y no haber información en la tabla de rutas que indique cómo llegar al destino D, se emplea el modo *broadcast* (proceso 2) para reenviar la RFA. El comportamiento del nodo E es similar al nodo B. Para simplificar no se representa el mismo en la figura. Llegados a este punto, S, C y F reciben el mensaje desde B. S lo descarta porque ya tiene la información al ser nodo

origen. Por su parte, C y F realizan el mismo proceso que hizo B, si bien, por razones de comodidad, no se indica el comportamiento de F en la figura. Finalmente, B y D reciben el mensaje de C (proceso 3). B lo descarta porque ya lo ha procesado. D, por su parte, lo procesa ya que es el nodo destino, enviando la correspondiente hormiga reactiva *hacia atrás* para establecer los valores de feromona entre el destino D y el origen S.

14.3.1.2 Proceso Reactivo Hacia Atrás

Al alcanzar el destino la RFA se convierte en una hormiga reactiva *hacia atrás* (*Reactive Backward Ant* (RBA)). Esta última sigue la lista de nodos visitados generada por RFA para volver al nodo origen s. En este proceso sólo se elige la primera copia de la hormiga *hacia adelante* que llega, descartando las restantes. De esta manera se establece una única ruta y, como se ha comentado anteriormente, se reduce la sobrecarga.

Como se ha comentado en la sección 12.3 de esta memoria, las hormigas artificiales se inspiran en las hormigas naturales, pero presentan unas capacidades adicionales que mejoran sus prestaciones. Así, mientras que las hormigas naturales depositan feromona tanto en la ida como en la vuelta, las hormigas artificiales disponen de una memoria interna donde almacenan la información de los nodos recorridos. Esta información se utiliza por las hormigas *hacia atrás* en la vuelta, razón por la cual el retorno de la hormiga al origen se hace en modo *unicast*. En este recorrido la hormiga *hacia atrás* se encarga de crear o actualizar un registro en la tabla de encaminamiento. Este registro almacena un *valor* que representa la inversa del coste en términos de tiempo estimado en ir un paquete de datos del nodo destino al origen pasando por los nodos intermedios.

La hormiga *hacia atrás* calcula incrementalmente una estimación o *coste* c_i^d del tiempo que tardaría un paquete de datos en viajar a través de esa lista P de nodos hacia el destino d partiendo del nodo i, actualizando las tablas de encaminamiento.

El proceso de actualización del registro de la tabla de encaminamiento es como sigue: la hormiga *hacia atrás* actualiza el número de saltos h_{in}^d y el valor de la feromona regular τ_{in}^d del registro de la tabla de encaminamiento, siendo n el nodo anterior visitado, i el nodo actual (el que se está procesando) y d el destino de la sesión.

La ecuación 14.3 resume el proceso de actualización del número de saltos h_{in}^d:

$$h_{in}^d \leftarrow \alpha h_{in}^d + (1-\alpha)h \qquad \alpha \in [0,1] \tag{14.3}$$

En esta ecuación h es el número de saltos que la hormiga *hacia atrás* ha recorrido y α un parámetro de regulación que indica cómo de rápido se adapta la fórmula a la nueva información. En los experimentos α se ha mantenido siempre en el valor habitual de 0,7.

El proceso de actualización de la feromona regular es como sigue:

La estimación c_i^d comentada anteriormente se calcula según la ecuación 14.4, esto es, viene a ser la suma de las estimaciones del tiempo que tarda en alcanzar el siguiente salto en cada nodo de la lista P:

$$c_i^d = \sum_{i=1}^{n-1} \hat{T}_{i \to i+1} \tag{14.4}$$

El valor de la estimación local $\hat{T}_{i \to i+1}$ se define como el producto de dos términos:

- El número actual de paquetes en la cola que están listos para enviarse en la capa MAC más uno, esto es:

$$Q_{mac}^i + 1$$

- El tiempo medio necesario para enviar un paquete

$$\hat{T}^i_{mac}$$

con lo que $\hat{T}_{i\to i+1}$ queda como indica la ecuación 14.5:

$$\hat{T}_{i\to i+1} = (Q^i_{mac} + 1)\hat{T}^i_{mac} \tag{14.5}$$

Si tenemos en cuenta el tiempo real t^i_{mac} que tarda un nodo en enviar un paquete:

$$\hat{T}^i_{mac} \leftarrow \eta\hat{T}^i_{mac} + (1-\eta)t^i_{mac} \qquad \eta \in [0,1] \tag{14.6}$$

En los experimentos η se ha establecido también a 0,7. Con este parámetro se quiere indicar que $\hat{T}^i_{mac}$ tiene más prioridad que t^i_{mac}, concretamente un 70 %. El valor t^i_{mac} en cada salto se calcula en AntOR como la diferencia de tiempo entre el envío y recepción de la hormiga *hacia atrás*.

Finalmente, la actualización del valor de feromona regular se calcula según lo indicado en la ecuación 14.7:

$$\tau^d_{ij} \leftarrow \gamma\tau^d_{ij} + (1-\gamma)(c^d_i)^{-1} \qquad \gamma \in [0,1] \tag{14.7}$$

Mediante la anterior ecuación el valor de un registro τ^d_{ij} de la tabla de encaminamiento del nodo i se actualiza, siendo j el siguiente salto, d el destino que se quiere alcanzar y γ un parámetro de ajuste que se ha establecido a 0,7 en los experimentos realizados.

En el caso particular de que exista feromona virtual en el enlace / arco que se quiere actualizar (como consecuencia de que el proceso de difusión se haya realizado antes que el proceso de establecimiento de ruta), esto es, si el nodo i que tiene una ruta al destino d usando el siguiente salto j ya tiene feromona virtual, la actualización de la feromona regular en el *proceso de establecimiento de ruta* descrito por la ecuación 14.7 queda, por el proceso de separación de feromonas regular y virtual, como sigue:

$$\begin{aligned} Regular_{final} &= F(Regular_{new}, time) \\ Virtual_{final} &= 0 \end{aligned} \tag{14.8}$$

siendo

$$Regular_{final} = F(Regular_{new}, time)$$

una representación simplificada de la ecuación 14.7.

En otras palabras, cuando se obtiene un nuevo valor de feromona regular el valor de feromona virtual se establece a 0. Se da prioridad, por tanto, a la feromona regular frente a la virtual, ya que los datos se encaminan solamente por rutas con valor de feromona regular. De esta forma no se origina ningún conflicto en la creación y mantenimiento (actualización) de las rutas estando el algoritmo optimizado respecto a su capacidad (memoria interna) debido a que la tabla de ruta sólo tiene una entrada por destino y siguiente salto. Esta entrada puede contener su correspondiente campo de feromona regular o virtual pero no ambas.

En la Figura 14.5 se muestra un ejemplo del proceso *hacia atrás* de actualización de la feromona y del número medio de saltos de la tabla de encaminamiento. La RBA va desde el nodo destino D hasta el nodo origen S según indican las flechas. En su camino de retorno se crean o actualizan los valores de la tabla de encaminamiento según la ecuación 14.7. El

proceso es el siguiente: D envía un mensaje *unicast* RBA al nodo C. En el instante en que C lo recibe crea la correspondiente entrada del registro en la tabla de encaminamiento.

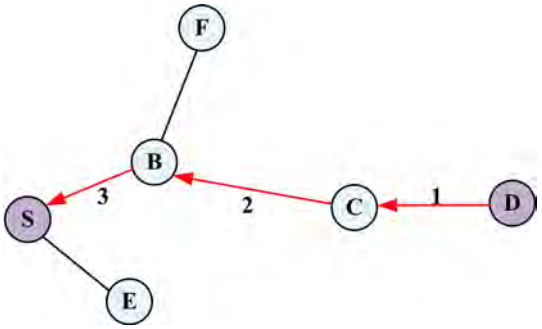


Figura 14.5: Ejemplo de asentamiento de la feromona (AntOR)

La información contenida en el registro es la siguiente:

- El destino
- El siguiente salto
- El valor de feromona regular
- El valor de la estimación del número medio de saltos

Los nodos B y S realizan la misma operación que C.

En la Figura 14.6 se muestra un esquema resumen del funcionamiento de la fase de establecimiento de ruta. Conviene reseñar que la creación de múltiples rutas a un mismo destino no se lleva a cabo en esta fase; tiene lugar en el proceso proactivo de exploración de rutas que se verá más adelante, concretamente en el apartado .

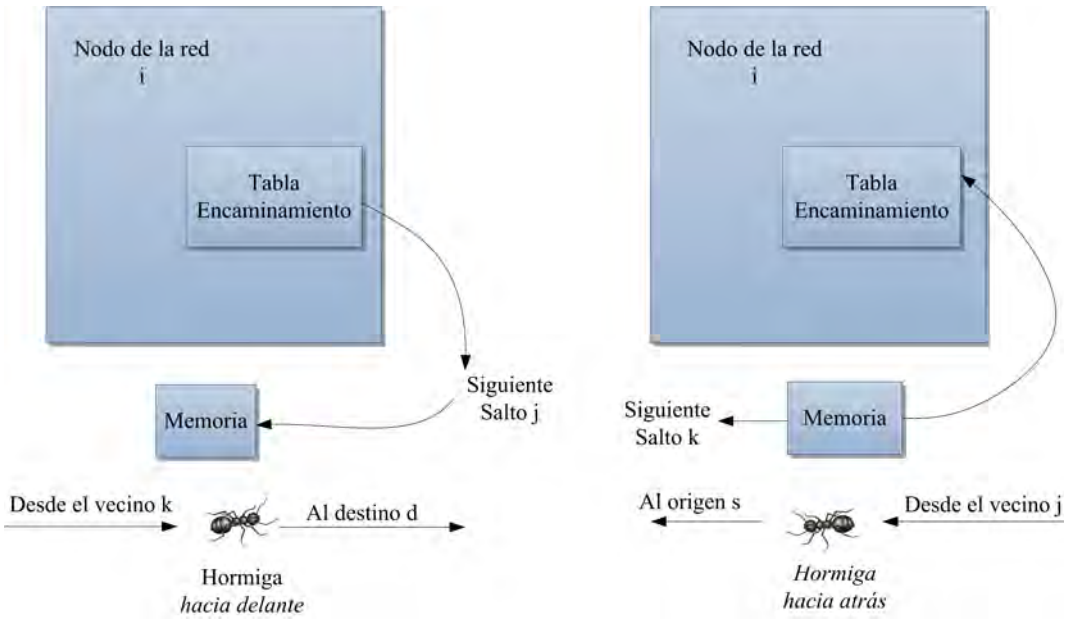


Figura 14.6: Funcionamiento del proceso de establecimiento de ruta (AntOR)

14.3.2 Encaminamiento Estocástico de los Datos

El primer establecimiento de ruta crea un camino único entre origen y destino, que se indica en la tabla de encaminamiento. Otros establecimientos de ruta y el proceso de exploración de ruta que se explica en el apartado siguiente originan múltiples caminos entre origen y destino. Esto conlleva que los datos pueden ser reenviados en modo multisalto de acuerdo a una técnica probabilística basada en las tablas de encaminamiento. La estrategia consiste en hacer que la carga de los datos se expanda mediante balanceo de carga. Esto es importante en las redes móviles ad hoc porque el ancho de banda del canal inalámbrico es muy limitado. El encaminamiento de los datos viene dado por la siguiente ecuación:

$$P_{in}^{d} = \frac{(\tau_{in}^{d})^{\beta_2}}{\sum_{j \in N_i^d} (\tau_{ij}^{d})^{\beta_2}} \qquad \beta_2 \geq 1 \tag{14.9}$$

La ecuación 14.9 es similar a la 14.1. La diferencia está en los parámetros exponenciales β_1 y β_2.

14.3.3 Mantenimiento de Rutas Establecidas y Exploración de Nuevas Rutas

Como su nombre indica, esta fase consiste en un proceso proactivo de mantenimiento de rutas establecidas y exploración de nuevas rutas, que actualiza y amplía la información disponible de encaminamiento. Esto permite construir múltiples rutas que sirven de respaldo a la ruta inicial creada en el proceso reactivo de establecimiento de ruta. Este proceso proactivo contempla dos subprocesos: *difusión de feromona* y *envío proactivo de hormigas*. Esta fase de AntOR difiere de la análoga de AntHocNet en la separación de las feromonas en el proceso de difusión, en la capacidad disjunta y en el uso de la métrica *distancia*, diferencias que inciden directamente en los dos subprocesos de esta fase.

14.3.3.1 Difusión de Feromona

En AntOR una ruta a un destino no puede tener simultáneamente un valor de feromona regular y de feromona virtual. Esta restricción o característica de AntOR no se contempla en AntHocNet y se denomina propiedad de separación de las feromonas.

La difusión de feromona tiene como objetivo expandir la información disponible de feromona en la red mediante el envío periódico de mensajes de actualización y la técnica de *bootstrapping* para conocer los destinos alcanzables de la red. Este proceso es similar a la difusión de la feromona en la naturaleza. Los mensajes *Hello* desempeñan un importante papel: cada cierto intervalo de tiempo t los nodos envían mensajes de este tipo en modo *broadcast*. En la experimentación se estableció t igual a 1 segundo. Estos mensajes se utilizan también para conocer los vecinos a 1-salto y detectar fallos de enlace. Al mismo tiempo estos mensajes sirven para difundir la feromona necesaria en el proceso de *bootstrapping*.

La generación de estos mensajes *Hello* es como sigue:

Un nodo i elige un número máximo k de destinos consultando la información de su tabla de encaminamiento. Cuando hay más destinos disponibles, éstos se seleccionan aleatoriamente. Experimentalmente se fijó el valor de k a 10 (para este valor se obtenían buenos resultados y sin apenas sobrecarga). Para cada destino d el mensaje *Hello* tiene información del mejor valor de feromona v_i^d que el nodo i tiene del destino d. Este se calcula teniendo en cuenta todos los posibles valores de feromona regular τ_{ij}^d y feromona

virtual ω_{ij}^d asociados al destino d. Además de incluirlo se indica con un *flag* si el valor elegido corresponde a un valor de feromona regular o virtual.

Una vez creado el mensaje *Hello* se envía, como se ha indicado anteriormente, en modo *broadcast*. Todos los vecinos del nodo a los que envía este mensaje *Hello* reciben una copia. Así pues, un nodo vecino j, al recibir este mensaje, estima un nuevo valor que indica cómo de *buena* es la ruta desde este nodo j al emisor i que tiene un destino alcanzable d indicado en la lista de destinos del mensaje *Hello*. Esta estimación se realiza combinando (*bootstrapping*) el valor de feromona v_i^d del mensaje *Hello* con la estimación local o *coste* c_j^i del salto j a i, es decir, del enlace entre el nodo j y el nodo i; c_j^i corresponde aquí a $\hat{T}_{i\to i+1}$ de la ecuación 14.4.

La ecuación 14.10 resume el proceso de *bootstrapping*:

$$k_{ji}^d = ((v_i^d)^{-1} + c_j^i)^{-1} \tag{14.10}$$

Siendo k_{ji}^d el valor *bootstrapped* obtenido en este proceso. Gracias al uso de esta técnica, la sobrecarga es baja, porque lo único que se necesita es enviar el valor v_i^d desde el nodo i al j. Esta información viene incluida en el mensaje *Hello* que se envía en *broadcast* y que cuando es recibido por un nodo nunca se reenvía (aumentaría la sobrecarga entonces).

Este proceso de *bootstrapping* se repite constantemente cuando comienza la simulación con el envío de los mensajes *Hello* de forma asíncrona por cada nodo de la red. Si bien este proceso presenta baja sobrecarga puede haber problemas de confiabilidad. El valor obtenido por *bootstrapping* sólo será correcto cuando lo sea el valor v_i^d incluido en el mensaje *Hello*. Esto resulta especialmente problemático en entornos altamente dinámicos donde la información de encaminamiento no se actualiza rápidamente y, especialmente, si el valor incluido en el mensaje *Hello* corresponde a *feromona virtual*. A esta problemática hay que añadirle el hecho de que el proceso de *bootstrapping* es relativamente lento, porque el envío de los mensajes *Hello* se efectúa cada cierto intervalo de tiempo (en aras de mantener su eficiencia).

Por las consideraciones anteriores, AntOR tiene la premisa de que el valor de feromona *bootstrapped* k_{ji}^d obtenido en la ecuación 14.10, es poco fiable. Esta característica influye directamente en la actualización de la tabla de encaminamiento cuando se usa este valor k_{ji}^d y en la separación de los valores de feromona regular y virtual. Generalmente, el valor de feromona virtual se actualiza con el nuevo valor de feromona *bootstrapped*. Por el contrario, sólo se actualiza la feromona regular por este valor de feromona bootstrapped (k_{ji}^d) cuando ocurren simultáneamente las siguientes condiciones:

a) El nodo j que recibe el correspondiente mensaje *Hello* tiene un valor de feromona regular distinto de cero.

b) Este mensaje *Hello* contiene también un valor v_i^d correspondiente a feromona regular.

Asimismo, AntOR aplica la siguiente premisa:

Si el nodo j que tiene una ruta al destino d ya tiene feromona regular y le llega feromona virtual contenida en el mensaje *Hello* durante el *proceso de difusión* de feromona, entonces el valor virtual no se actualiza en el nodo j, puesto que no puede haber simultáneamente valores no nulos en ambas feromonas. El valor de feromona virtual final es, por tanto, cero, tal y como recoge la ecuación 14.11.

$$\begin{aligned} Regular_{final} &= Regular_{old} \\ Virtual_{final} &= 0 \end{aligned} \tag{14.11}$$

Como se ha explicado antes en la subsección 14.2.1, otra ventaja que se obtiene al aplicar esta restricción de separación de las feromonas es que la feromona virtual se comporta como un valor de respaldo (alternativo) en la exploración de nuevas rutas, mientras que los datos sólo sean encaminados por rutas de feromona regular.

Asimismo, conviene señalar que las hormigas de difusión de feromona también sirven para detectar enlaces rotos. Así, cuando falla un enlace, los nodos pueden actualizar sus tablas de encaminamiento.

Para comprender mejor lo explicado en este apartado, la Figura 14.7 muestra un ejemplo de selección del mejor valor de feromona. Las flechas en color rojo indican enlaces de feromona regular y la de color negro un enlace de feromona virtual. El nodo actual A crea su correspondiente mensaje *Hello* seleccionando un destino alcanzable (D en este caso) y eligiendo el mejor valor de feromona entre los valores disponibles ω_{AC}^{D} y τ_{AB}^{D}.

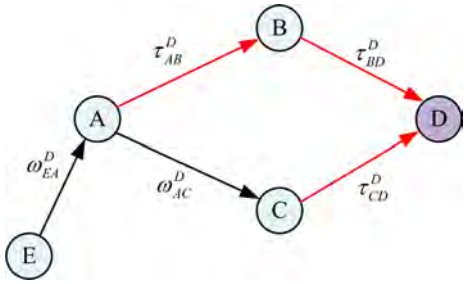


Figura 14.7: Ejemplo de selección del mejor valor de feromona (AntOR)

Supóngase que se elige el valor de feromona virtual ω_{AC}^{D} como valor v_{A}^{D}. A continuación se procede al envío del mensaje *Hello* en modo *broadcast*. Supóngase ahora que el nodo E recibe el mensaje *Hello* enviado por A. El nuevo valor *bootstrapped* del enlace entre E y A con destino alcanzable D se obtiene aplicando la siguiente fórmula:

$$k_{EA}^{D} = ((v_{A}^{D})^{-1} + c_{E}^{A})^{-1} \tag{14.12}$$

Según la ecuación 14.12 el nodo E obtiene un nuevo valor de feromona combinando la información del mensaje *Hello* con el coste estimado del enlace entre E y su vecino A del cual recibe el mensaje. Las inversiones de esta ecuación son necesarias primero para convertir un valor de feromona en un valor compatible con el *coste* c_{E}^{A} y así poder realizar la suma, y segundo para volver a convertir la suma total en un valor de feromona. Este valor final k_{EA}^{D} puede utilizarse para actualizar ω_{EA}^{D} y τ_{EA}^{D}. Como no se cumple la premisa de la ecuación 14.11 se actualiza ω_{EA}^{D}.

14.3.3.2 Envío Proactivo de Hormigas

Este subproceso implica la exploración de nuevas rutas mediante el envío de hormigas proactivas. Este subproceso tiene en cuenta la propiedad disjunta de enlace/nodo, por lo que se procede, en primer lugar, a comentar las características generales de este tipo de rutas, para seguidamente explicar el funcionamiento de la exploración de nuevas rutas basándose en la métrica *distancia*.

14.3.3.3 Rutas Disjuntas de Enlace/Nodo

Las rutas de enlace/nodo disjunto (también llamadas rutas disjunta de enlace/nodo) son rutas que no comparten enlaces/nodos, respectivamente. Las rutas de enlace disjunto

son menos restrictivas y más fáciles de calcular, si bien tienen un menor nivel de tolerancia ante los fallos que las rutas de nodo disjunto. Se cumple la propiedad de que toda ruta de nodo disjunto es también de enlace disjunto, pero no al contrario. Ambos tipos de rutas presentan las siguientes ventajas:

- Un fallo en un nodo sólo afecta a una ruta, no a toda la red.
- El balanceo de carga es mejor porque no se repiten rutas debido a la propiedad disjunta.

No obstante, la utilización de este tipo de rutas presenta los siguientes inconvenientes:

- Se necesitan más recursos porque no comparten enlaces/nodos.
- Estas rutas son más difíciles de descubrir, porque quedan limitados los nodos que pueden ser visitados.

Existen dos modalidades de creación de rutas disjuntas, denominadas A y B.

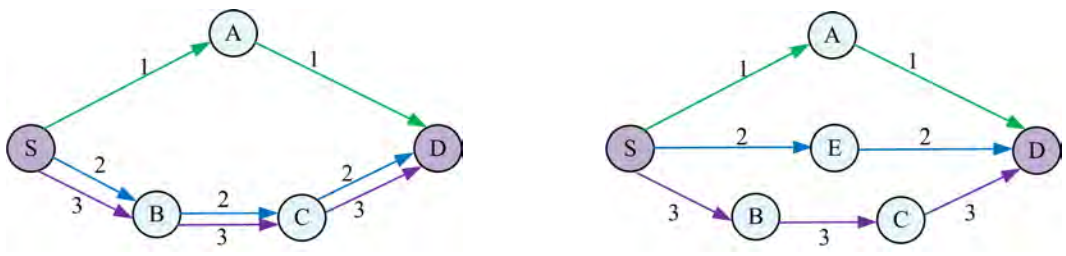
En la primera (modalidad A), una vez creada la ruta principal por el establecimiento de ruta y cuando se procede a descubrir otras nuevas, hay que tener presente la premisa de que estas rutas son disjuntas respecto a la principal, por lo que estas alternativas se pueden ser repetir. Esto hace que sea una modalidad menos restrictiva, pudiendo actualizarse con más frecuencia las tablas de encaminamiento, si bien conlleva una mayor sobrecarga por el envío de más agentes proactivos.

La segunda (modalidad B) crea una ruta principal y otras rutas alternativas todas disjuntas entre sí. Esta modalidad es más restrictiva pero ocasiona una menor sobrecarga.

La modalidad B tiene una mejor tolerancia ante un fallo de enlace que la modalidad A porque posibilita un mayor número de rutas disjuntas, presentando el inconveniente de que las rutas alternativas pueden no estar actualizadas en ese momento.

Tras diversos experimentos se optó en las simulaciones por la modalidad B por las consideraciones expuestas: menor sobrecarga y mayor gama de rutas disjuntas.

En la Figura 14.8 puede verse la comparativa de las dos modalidades de cálculo de rutas disjuntas. La ruta principal está etiquetada con el número 1 y las rutas alternativas que se calculan con los números 2 y 3. Se observa cómo en la modalidad A una ruta alternativa ya creada (número 2) se recalcula por segunda vez, al mismo tiempo que se actualizan las tablas de feromona de los nodos que la constituyen (número 3). Esto mantiene las rutas a costa de una mayor sobrecarga, ya que se envían más hormigas proactivas de exploración de rutas. Por otro lado, la red asociada a la modalidad B está formada por rutas disjuntas entre sí, lo que produce menor sobrecarga, mejor tolerancia ante fallos de enlace y menor frecuencia de actualización de rutas.



(a) Modalidad A (b) Modalidad B

Figura 14.8: Comparativa de las dos modalidades (AntOR)

La Figura 14.9 ilustra gráficamente el concepto de rutas disjuntas de enlace. La Figura 14.9(a) representa una ruta disjunta de enlace, donde el nodo A no comparte enlaces. Se observa que los nodos S y A sólo comparten un único enlace (S-A). Por el contrario, la Figura 14.9(b) representa la versión de ruta no disjunta de enlace porque los nodos S y A comparten dos enlaces solapados de la misma sesión de datos.

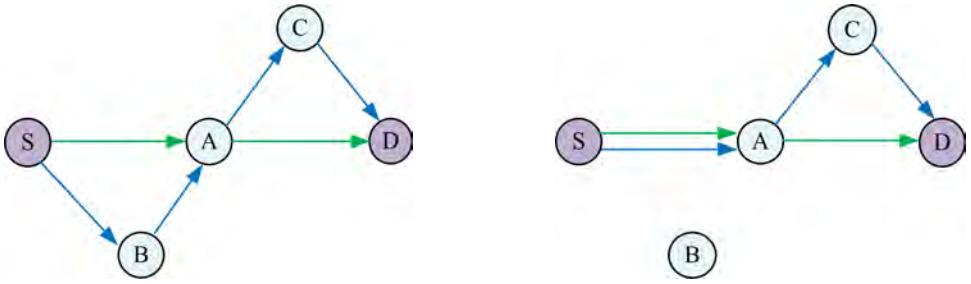


Figura 14.9: Rutas de enlaces disjuntos a) versus no disjuntos b) (AntOR)

La Figura 14.10 ilustra el concepto de rutas disjuntas de nodo. En la Figura 14.10(a) podemos observar dos rutas: la principal (color verde) y la alternativa (color azul). Estas rutas no comparten nodos, considerándose disjuntas de nodo. Por el contrario, la Figura 14.10(b) muestra el caso de rutas no disjuntas de nodo ya que en el nodo A se solapan dos rutas: la principal y la alternativa.

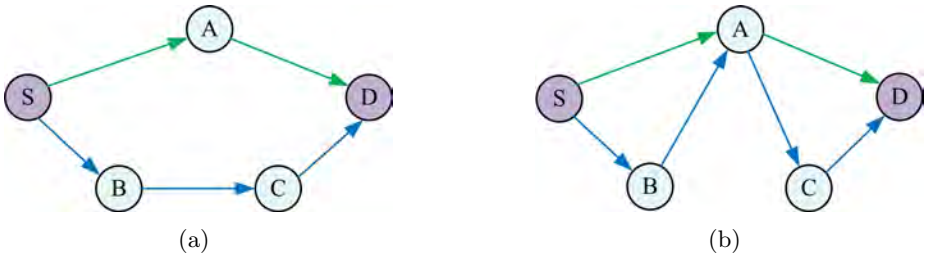


Figura 14.10: Rutas de nodos disjuntos a) versus no disjuntos b) (AntOR).

Las Figuras 14.11 y 14.12 ilustran una comparativa gráfica entre las rutas disjuntas de enlace y de nodo. En ellas se presenta un escenario formado por 6 nodos, en el que el nodo origen A tiene dos rutas creadas (verde y azul) al destino F.

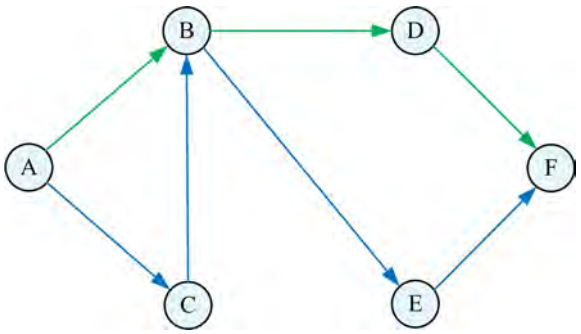


Figura 14.11: Escenario: Ruta de enlace disjunto (AntOR)

La Figura 14.11 corresponde a un ejemplo de ruta disjunta de enlace y la Figura 14.12 a otro de ruta disjunta de nodo. En el citado escenario acontecen dos fallos de enlace/nodo:

a) Falla el envío del mensaje entre el enlace (B-D)

b) Falla el nodo B (sale del rango de cobertura o se deshabilita)

En el caso a) el escenario de la Figura 14.11 tiene una ruta alternativa formada por el enlace (B-E) para el envío de la información. Sin embargo, en el escenario de la Figura 14.12 se tiene que realizar un proceso de neutralización de fallo de enlace porque se observa que en las rutas disjuntas de nodo los nodos intermedios no pueden tener caminos alternativos.

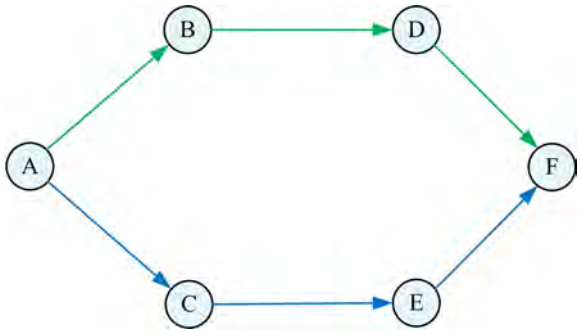


Figura 14.12: Escenario: Ruta de nodo disjunto (AntOR)

En el caso b) el escenario de la Figura 14.12 ofrece una mejor tolerancia ante el fallo que el de la Figura 14.11 porque emplea la segunda alternativa (azul) para enviar el mensaje al nodo C (a través del enlace A-C). Por otro lado, en el escenario de la Figura 14.11 se aprecia el hecho de que si desaparece o se deshabilita el nodo B se rompen las dos rutas (verde y azul), haciendo imposible la comunicación con el destino F hasta que ocurra otro establecimiento de ruta.

Este sencillo ejemplo explica cómo las rutas disjuntas de nodo son más restrictivas, más tolerantes ante los fallos, más difíciles de calcular y, en algunos escenarios particulares, pueden presentar un comportamiento peor que las disjuntas de enlace. También con este ejemplo queda claro que toda ruta disjunta de nodo es también disjunta de enlace, pero no al contrario.

14.3.3.4 Funcionamiento

La exploración consiste en un proceso para descubrir nuevas rutas que sirvan como alternativas para el envío de los paquetes de datos. El proceso de difusión comentado anteriormente es imprescindible para el correcto funcionamiento de la exploración. El nodo origen de una sesión inicia este proceso de exploración en el momento en que el nodo destino recibe el primer paquete de datos de una nueva sesión. Este proceso se mantiene mientras esté activa la sesión. Inicialmente se genera la correspondiente hormiga proactiva *hacia adelante* (PFA) para su posterior envío. Estas hormigas nunca se envían en modo *broadcast*, ya que ellas sólo van por caminos que tienen marcada la ruta, bien por feromona regular, bien por feromona virtual.

En AntHocNet [Duc07], en aras de eficiencia, sólo se envía una hormiga proactiva *hacia adelante* si el mejor valor de feromona virtual es superior (al menos en un 10 %) al correspondiente de feromona regular. Esta característica no se aplica en AntOR debido a

que la propiedad disjunta y la métrica *distancia* limitan el envío de hormigas proactivas *hacia adelante*, lo que supondría una hipótesis más restrictiva. En AntHocNet [Duc07] la ecuación de exploración de rutas queda como sigue:

$$P_{in}^{d} = \frac{[max(\tau_{in}^{d}, \omega_{in}^{d})]^{\beta_3}}{\sum_{j \in N_i^d}[max(\tau_{ij}^{d}, \omega_{ij}^{d})]^{\beta_3}} \qquad \beta_3 \geq 1 \tag{14.13}$$

En AntOR la ecuación de exploración queda como sigue:

$$P_{in}^{d} = \frac{(\psi_{in}^{d})^{\beta_3}}{\sum_{j \in N_i^d}(\psi_{ij}^{d})^{\beta_3}} \qquad \psi \in \begin{cases} \omega & virtual \\ \tau & regular \end{cases} \tag{14.14}$$

donde ψ es un valor de feromona regular o virtual y β_3 un parámetro de ajuste relativo a la influencia de la concentración de feromona (con funcionalidad análoga a la de β_1 y β_2).

Conviene reseñar que en AntOR se utiliza la métrica *distancia*, circunstancia que no ocurre en AntHocNet [Duc07]. Así, se considera el número de saltos de las mejores rutas halladas. De esta forma se controla que una hormiga proactiva no pueda recorrer más nodos de los establecidos por el denominado límite de saltos, que se establece según las mejores rutas (aquellas con menor distancia en número de saltos) calculadas anteriormente. La razón de elección de esta métrica (y no otras, como el retardo por ejemplo) es que se considera estable, puesto que no le influyen las interferencias producidas por otros dispositivos. Las PFAs llegan a sus destinos convirtiéndose en hormigas proactivas *hacia atrás* (PBA). Estas últimas tienen una funcionalidad de actualización de las tablas de encaminamiento análoga a la comentada en las RBAs del apartado 14.3.1.2.

La Figura 14.13 ilustra un ejemplo de funcionamiento de esta etapa de exploración de rutas en lo relativo a la separación de las feromonas.

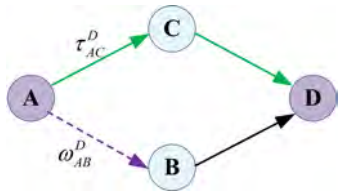


Figura 14.13: Ejemplo de exploración de rutas (AntOR)

En el ejemplo la ruta alternativa 1 (línea verde continua) tiene sólo feromona *regular* y la ruta alternativa 2 (línea morada discontinua) sólo feromona *virtual*. La probabilidad de elección de la ruta alternativa 1 viene dado por la siguiente ecuación:

$$P_{AC}^{D} = \frac{(\tau_{AC}^{D})^{\beta_3}}{(\tau_{AC}^{D})^{\beta_3} + (\omega_{AB}^{D})^{\beta_3}} \tag{14.15}$$

La Figura 14.14 ilustra un ejemplo del uso de la métrica *distancia*. Se trata de un escenario de ruta de nodo disjunto. Inicialmente, se crea la ruta principal, representada en color verde y etiquetada con M. Para el envío de la correspondiente hormiga proactiva en el proceso de exploración de nuevas rutas se elige una de las dos alternativas posibles: ruta alternativa A_1 o ruta alternativa A_2. Esta elección está basada en el *coste* (*distancia*) que supone llegar del nodo origen S al destino D. En este ejemplo la ruta alternativa más propicia (ruta candidata) para transmitir (para enviar la PFA) es A_1 con siguiente salto

C. Puede comprobarse cómo la alternativa A_1 tiene 2 saltos frente a los 4 de la alternativa A_2 y a los 3 de la ruta principal M.

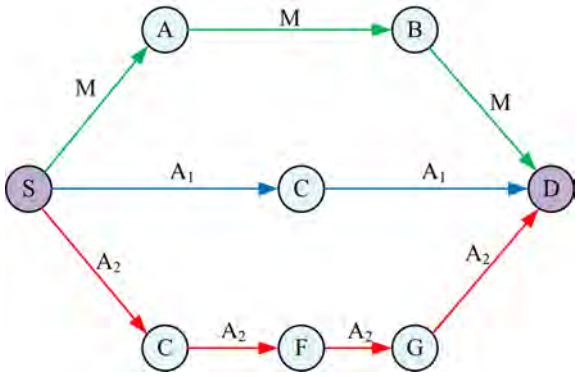


Figura 14.14: Esquema de exploración usando la métrica *distancia* (AntOR)

14.3.4 Gestión de Fallos de Enlace

Los nodos pueden detectar fallos de enlace en una transmisión *unicast* o cuando se espera un mensaje *Hello* y no se recibe. Cuando un enlace falla, el nodo puede perder la ruta a uno o más destinos. Un ejemplo de fallo de enlace se produce cuando un vecino se mueve más allá del rango de transmisión. En el fallo de enlace se consideran dos clases de problemas:

- Si el nodo tiene otras alternativas al destino o si la ruta al destino se ha perdido porque no se ha usado regularmente, se tiene que notificar con un mensaje de fallo de enlace. Así, el nodo actualiza su tabla de encaminamiento y envía una hormiga de *notificación de fallo* en modo *broadcast*. Esta hormiga contiene una lista de los destinos que perdieron el camino: el nuevo retardo extremo a extremo estimado y el número de saltos a este destino. Todos sus vecinos reciben la notificación y actualizan su tabla de feromona usando las nuevas estimaciones. Por otro lado, si los vecinos pierden su mejor o su único camino a un destino debido al fallo, generan y envían una hormiga de fallo en modo *broadcast*, hasta que todos los nodos de los diferentes caminos hayan recibido notificación de la nueva situación.

- Si se pierde la ruta a un destino regularmente usado por los datos y es la única alternativa del nodo, la pérdida es especialmente importante y el nodo intenta reparar localmente el camino. En AntOR el nodo sólo repara el camino si descubre que el enlace perdido es debido al fallo de una transmisión de paquetes de datos.

La Figura 14.15 muestra un esquema del proceso de neutralización de los fallos de enlace en AntOR.

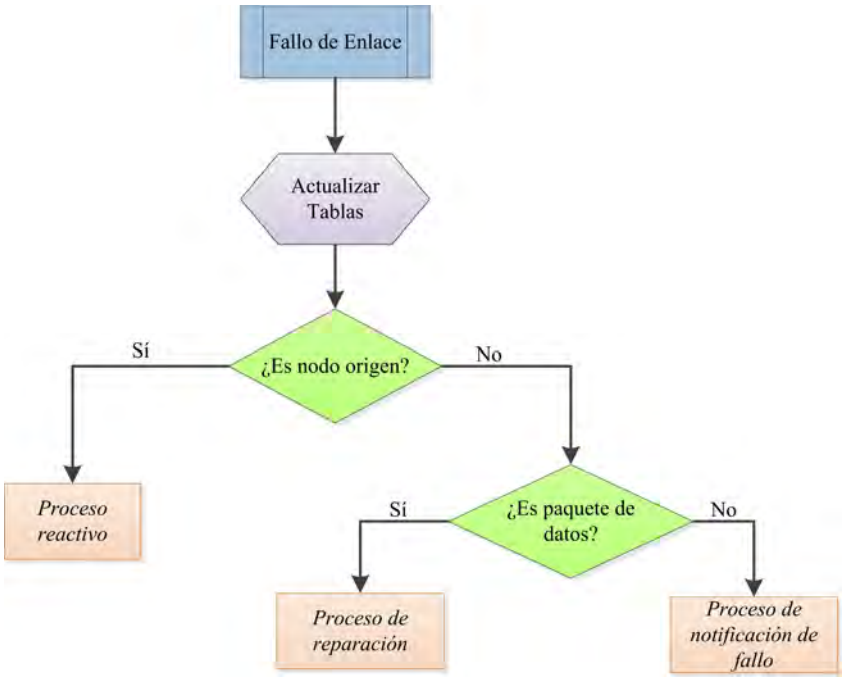


Figura 14.15: Gestión fallos de enlace (AntOR)

Lo primero que ocurre cuando hay un fallo de enlace es que el nodo que lo detecta lo elimina de su tabla de vecinos. A continuación, se actualiza la tabla de encaminamiento con la nueva información de feromona. Por último, se encarga de neutralizar el fallo teniendo en cuenta los siguientes factores:

a) Si no hay ruta en el origen, se envía una hormiga reactiva *hacia adelante*.

b) Si no hay ruta en un nodo intermedio y se trata de un paquete de datos lo que se estaba retransmitiendo cuando se produjo el fallo, se envía una hormiga *hacia adelante* de reparación de ruta. Si no hay respuesta de la correspondiente hormiga *hacia atrás* de reparación en un determinado período de tiempo, se envía en modo *broadcast* un mensaje de notificación de fallo de enlace, informando que el destino es inalcanzable.

c) Cuando hay un fallo de enlace, debido a que no recibe el correspondiente mensaje *Hello* consecutivo en un determinado tiempo o porque se pierde un mensaje de control *unicast* en algunos de los nodos intermedios, se crea un mensaje de notificación de fallo de enlace informando de los destinos inalcanzables enviándose en modo *broadcast*.

Uno de los mecanismos de neutralización presentado en la Figura 14.15 es el proceso de reparación de ruta que es muy similar a un establecimiento de ruta: el nodo envía una hormiga de reparación de ruta *hacia adelante* (*Route Repair Forward Ant* (RRFA)) en modo *broadcast* y los nodos intermedios reenvían esta hormiga del mismo modo, pero con un límite máximo de intentos. Sin embargo, si hay información de ruta disponible en los nodos intermedios, el envío a través de ellos se efectúa en modo *unicast*, aplicando la ecuación 14.1.

En este proceso de reparación de ruta se necesita la información del número de saltos h_{ij}^{d} que se encuentra en la tabla de encaminamiento, puesto que el nodo que inicia el

proceso de reparación espera a que le llegue una hormiga de reparación de ruta *hacia atrás* (*Route Repair Backward Ant* (RRBA)) un cierto tiempo:

$$T_{Espera} = 2t_{hop}h_{ij}^{d} \tag{14.16}$$

La ecuación 14.16 representa una estimación del tiempo que tarda en ir y volver del nodo i al nodo destino d. Se establece t_{hop}, el valor fijo de retardo por salto, a 50 ms.

Si no recibe la correspondiente RRBA da por terminado el proceso, al no reparar la ruta en el tiempo establecido por la ecuación 14.16. Consecuentemente, el nodo que detecta el fallo descarta el paquete de datos previamente encolado (porque no se ha reparado correctamente la ruta según el temporizador), generándose una hormiga de notificación de fallo que anuncia la nueva situación.

La Figura 14.16 ilustra un proceso de reparación de ruta.

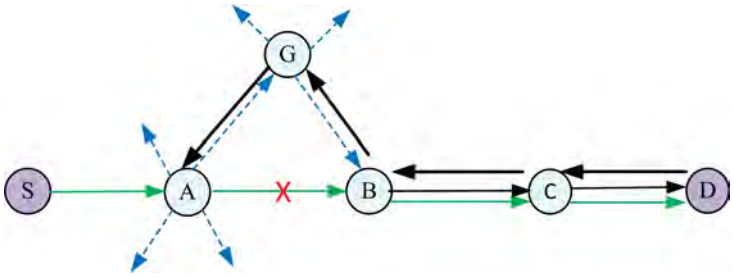


Figura 14.16: Ejemplo de reparación local de ruta (AntOR)

En este ejemplo la ruta de la sesión de datos entre el nodo origen S y el destino D (línea de color verde y camino S-A-B-C-D) está rota como consecuencia de un fallo de enlace entre los nodos A y B. El nodo A que detecta el fallo intenta repararlo enviando una RRFA hacia el destino D en modo *broadcast* (la ruta marcada con una línea discontinua en color azul). G recibe una copia de esta hormiga y la difunde. A continuación, B recibe la RRFA y la envía en modo *unicast*, porque hay ruta (color negro) entre este nodo y el destino D que constituye la parte de ruta original que es válida. Finalmente, el nodo destino D envía una RRBA al nodo local A (camino D-C-B-G-A). Esta hormiga, al igual que en el proceso reactivo, se encarga de actualizar las tablas de encaminamiento de los nodos visitados en la ruta de vuelta.

14.3.5 Resumen

Las Figuras 14.17 a 14.19 muestran un ejemplo completo del funcionamiento de AntOR. En el escenario de la Figura 14.17 se observa que hay una única sesión de datos formada por el par (A, D); en otras palabras, A envía la información al destino D.

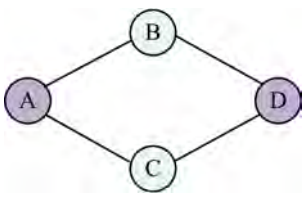


Figura 14.17: Ejemplo de marcado y asentamiento de ruta (AntOR)

Se entiende por iniciar la sesión de datos el hecho de que se quiere enviar datos desde el origen S al destino D. Con los mensajes *Hello* se crean rutas independientes de las sesiones de datos entre pares de vecinos que están a un salto. En el ejemplo se muestran 8 rutas: A-C, C-A, A-B, B-A, B-D, D-B, C-D, D-C.

La Figura 14.18 ilustra el proceso de marcado y asentamiento de ruta para el cálculo de τ y h. Esto se realiza según lo visto en el apartado 14.3.1.2. En este ejemplo el valor de feromona virtual ω vale 0 para todas las rutas.

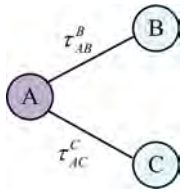


Figura 14.18: Ejemplo de escenario del funcionamiento del protocolo (AntOR)

Las tablas de rutas (de la 14.5 a la 14.8) quedan de la siguiente forma:

Tabla 14.5: Rutas para el nodo A (AntOR)

Rutas para Nodo A	**Destino**	**Siguiente Salto**	τ	ω	**h**
Vecinos	B	B	0,3	0	0,3
	C	C	0,3	0	0,3

Tabla 14.6: Rutas para el nodo D (AntOR)

Rutas para Nodo D	**Destino**	**Siguiente Salto**	τ	ω	**h**
Vecinos	B	B	0,3	0	0,3
	C	C	0,3	0	0,3

Tabla 14.7: Rutas para el nodo B (AntOR)

Rutas para Nodo B	**Destino**	**Siguiente Salto**	τ	ω	**h**
Vecinos	A	A	0,3	0	0,3
	D	D	0,3	0	0,3

Tabla 14.8: Rutas para el nodo C (AntOR)

Rutas para Nodo C	**Destino**	**Siguiente Salto**	τ	ω	**h**
Vecinos	A	A	0,3	0	0,3
	D	D	0,3	0	0,3

Otro proceso que se realiza independientemente de la sesión de datos es el *proceso de difusión*. Supóngase el siguiente caso: el nodo A todavía no tiene ruta a D a través de C, así que el nodo C difunde la información de su destino D al nodo A. El nodo C informa al nodo A de la mejor ruta al destino D; en este caso sólo tiene una. Si hubiera otras alternativas elegiría el mejor valor *regular* o *virtual* al destino D.

La Figura 14.19 muestra el proceso de difusión de feromona donde la línea discontinua representa la feromona *virtual* cuyo valor (0,3) está en función del mejor destino de C a D.

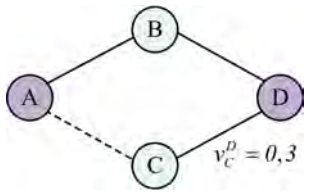


Figura 14.19: Esquema de difusión de encaminamiento (AntOR)

El nodo A, que recibe la información de C, aplica la ecuación 14.10, quedando la nueva tabla de rutas de este nodo tal y como se indica en la Tabla 14.9. Si se diera el caso de que A tuviera un valor regular el valor virtual no se actualizaría en el nodo A. Este proceso de *difusión de encaminamiento* se repite constantemente cada cierto tiempo.

Tabla 14.9: Rutas para A en el proceso de difusión (AntOR)

Rutas para Nodo A	**Destino**	**Siguiente Salto**	τ	ω	**h**
Vecinos	B	B	0,3	0	0,3
	C	C	0,3	0	0,3
	D	C	0	0,23	0

Para finalizar se ha creído conveniente señalar un esquema general del funcionamiento de AntOR. La Figura 14.20 muestra tal esquema.

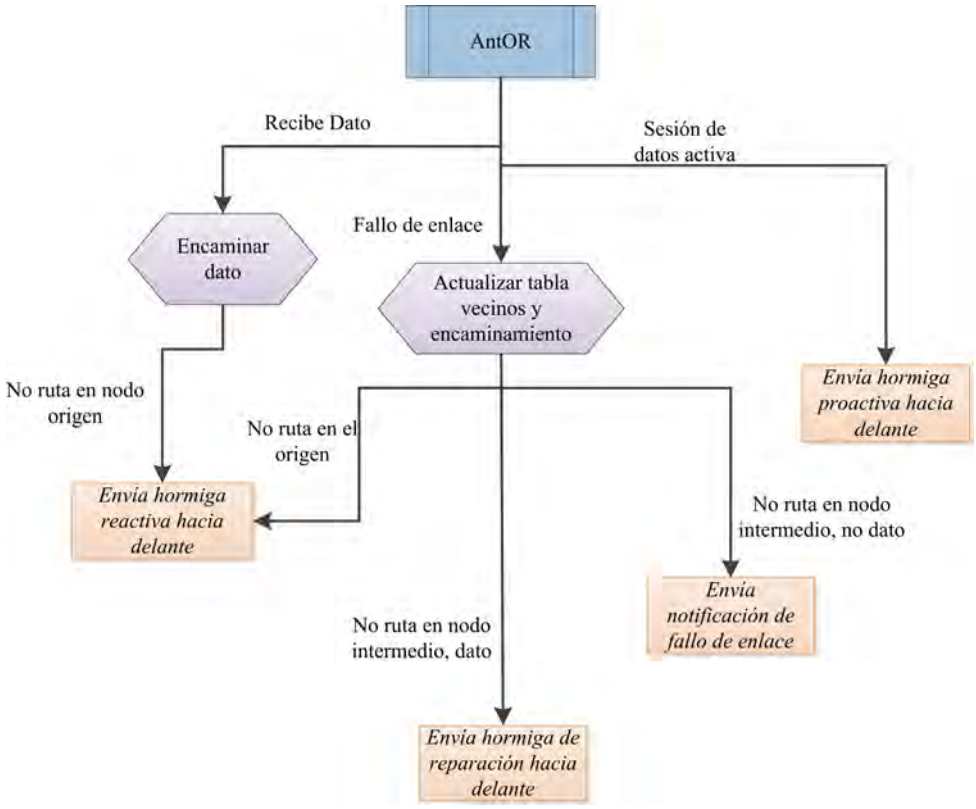


Figura 14.20: Diagrama funcionamiento de AntOR

14.4 AntOR - Disjoint Link Route (AntOR-DLR)

Como su nombre indica, *AntOR - Disjoint Link Route* (AntOR-DLR) se deriva del protocolo base AntOR con la única restricción de que en su especificación sólo tiene en cuenta rutas que no comparten enlaces. La Tabla 14.10 muestra la tabla de encaminamiento de AntOR-DLR. Como puede observarse, la tabla de encaminamiento añade respecto a AntOR un campo adicional denominado Sesión Disjunta. La Figura 14.21 muestra un esquema de cómo se constituyen las rutas de enlace disjunto. La idea básica para hallar y representar rutas de enlace disjunto consiste en *marcar* cada enlace disjunto con una etiqueta que indique cuál es el origen de la sesión de datos. Esta *marca* se indica en el campo *Sesión Disjunta* de la tabla de encaminamiento comentado anteriormente.

Tabla 14.10: Tabla de encaminamiento (AntOR-DLR)

Valores / Entradas	**Destino**	**Siguiente Salto**	**Valor de Feromona Regular (τ)**	**Valor de Feromona Virtual (ω)**	**Número medio de saltos (h)**	**Sesión disjunta (o)**
$Entrada_1$	$Destino_1$	$Siguiente\ Salto_1$	τ_1	ω_1	h_1	o_1
$Entrada_2$	$Destino_2$	$Siguiente\ Salto_2$	τ_2	ω_2	h_2	o_2
...	...	...	...	...	...	...
$Entrada_i$	$Destino_i$	$Siguiente\ Salto_i$	τ_i	ω_i	h_i	o_i
...	...	...	...	...	...	...

En la Figura 14.21 se observa una red compuesta por 5 nodos y dos rutas disjuntas (la ruta principal de color verde y la alternativa de color rojo y con línea discontinua). En ambas rutas se *marcan* los enlaces en el campo *Sesión Disjunta* con el origen de la sesión de datos. Por ejemplo, el nodo A tiene dos entradas en su tabla de encaminamiento para los enlaces (A, C) y (A, B) con, entre otras, la siguiente información: *destino*, *siguiente salto* y *sesión disjunta*. Para la *entrada 1* se tiene la combinación (D, C, A) y para la *entrada 2* la combinación (D, B, A).

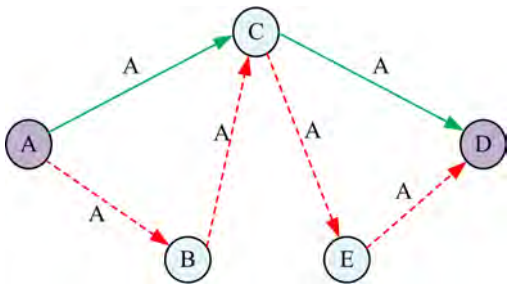


Figura 14.21: Esquema representativo de rutas disjuntas de enlace (AntOR-DLR)

La Figura 14.22 muestra el diagrama de flujo del procedimiento de cálculo de las rutas disjuntas de enlace. Como puede observase, el procedimiento es como sigue: se consulta el campo *Sesión Disjunta* en la tabla de encaminamiento para comprobar si el enlace ya es disjunto o no. Para ello se comprueba si el enlace Link está asociado con el origen de la sesión de datos. En cado negativo se envía la correspondiente hormiga proactiva *hacia adelante* al siguiente salto calculado anteriormente. Al recibir esta hormiga proactiva el proceso se repite en los nodos intermedios.

Figura 14.22: Diagrama flujo calculo de rutas disjuntas de enlace (AntOR-DLR)

El Algoritmo 14.1 representa el pseudocódigo del proceso de cálculo de rutas de enlace disjunto.

```
1 mientras Proceso proactivo hacer
2     {src,dst} = ObtenerInfoSesion(msg);
3     {nexthop} = ObtenerSiguienteSalto(dst);
4     {link} = ObtenerEnlace(dst,nexthop);
5     si ComprobarEnlaceDisjunto(link) = FALSO entonces
6         Enviar(nexthop,msg);
7     fin
8 fin
```

Algoritmo 14.1: Cálculo de rutas de enlace disjunto (AntOR-DLR)

En la línea 1 se muestra un proceso proactivo que está representado con un bucle para indicar que se realiza continuamente después del comienzo de la sesión de datos. En las líneas 2 y 3 se obtiene la información de la sesión datos (origen y destino) y el siguiente salto asociado al destino, respectivamente. En la línea 4 se obtiene el enlace *link* buscando en la tabla de encaminamiento del nodo local. En la línea 5 el método *ComprobarEnlaceDisjunto* comprueba la propiedad disjunta de enlace, es decir, si este enlace link calculado anteriormente coincide con el origen de la sesión de datos indicado en la PFA. En caso negativo (no hay ruta disjunta de enlace), se envía la hormiga proactiva (mensaje *msg*) al siguiente salto (*nexthop*) previamente calculado.

La Figura 14.23 muestra un ejemplo de cálculo de rutas disjuntas. En el citado ejemplo sólo hay una sesión de datos formada por el nodo origen S y el nodo destino D. Primero se calcula la ruta disjunta S1 marcando cada enlace disjunto con el origen de la sesión S. Después se calcula la ruta alternativa S2 teniendo en cuenta no repetir enlaces que ya pertenecen al origen S. La característica de poder compartir nodos hace que se puedan visitar nodos intermedios por rutas alternativas, pero no los enlaces que pertenecen a otras rutas.

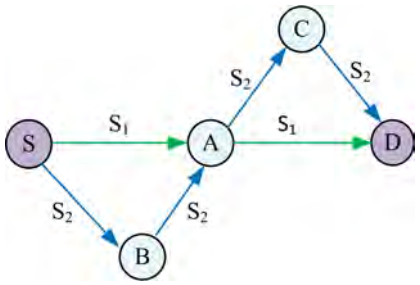


Figura 14.23: Ejemplo I: Una sesión de datos (AntOR-DLR)

Cuando se trabaja con varias sesiones de datos, varios pares origen/destino, la propiedad disjunta también se cumple, porque el marcado de las rutas es único para cada ruta disjunta perteneciente a cada sesión de datos. La Figura 14.24 muestra un ejemplo en el que se solapan rutas disjuntas pertenecientes a sesiones de datos diferentes. Se observa que hay dos sesiones de datos formadas por los pares (B-E) y (A-D), comprobándose que, aunque haya solapamiento, al ser sesiones distintas de datos, no se altera el comportamiento del protocolo.

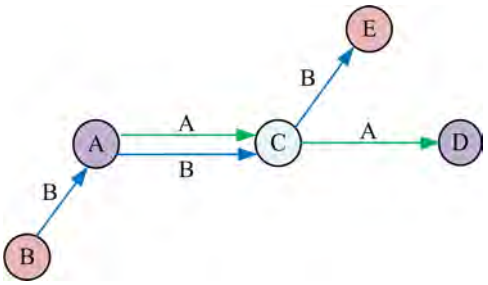


Figura 14.24: Ejemplo II: Dos sesiones de datos (AntOR-DLR)

14.5 AntOR - Disjoint Node Route (AntOR-DNR)

Como su nombre indica, *AntOR - Disjoint Node Route* (AntOR-DNR) se deriva del protocolo base AntOR con la única restricción de que en su especificación sólo tiene en cuenta rutas que no comparten nodos. Al igual que AntOR-DLR, la tabla de encaminamiento de AntOR-DNR añade respecto a AntOR un campo adicional denominado Sesión Disjunta.

La diferencia principal entre AntOR-DNR y AntOR-DLR consiste en la manera de calcular las rutas en el proceso exploratorio: en las rutas disjuntas de nodo es el nodo el encargado de detectar la propiedad disjunta, mientras que en las rutas disjuntas de enlace es el propio enlace.

La Figura 14.25 muestra un esquema de cómo se constituyen las rutas de nodo disjunto. Se observa una red formada por 5 nodos, una ruta principal de color verde y una posible ruta alternativa disjunta de color rojo y de trazo discontinuo.

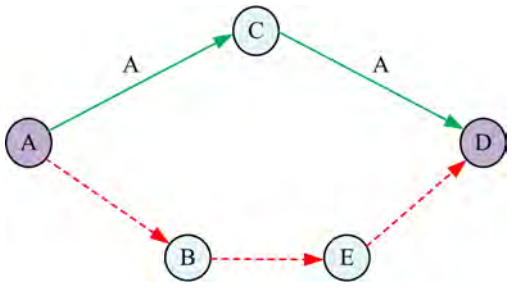


Figura 14.25: Esquema representativo de rutas disjuntas de nodo (AntOR-DNR)

La Figura 14.26 contiene el diagrama de flujo del funcionamiento de AntOR-DNR. El protocolo trabaja como sigue. Inicialmente, se envía la correspondiente hormiga proactiva *hacia adelante* al siguiente salto aplicando la ecuación 14.14. Cuando el nodo recibe la hormiga, consulta en su tabla de encaminamiento si el campo sesión disjunta tiene el mismo valor que el origen de la hormiga. En el caso de que tengan el mismo valor se descarta el paquete, porque se trata de una ruta disjunta de nodo.

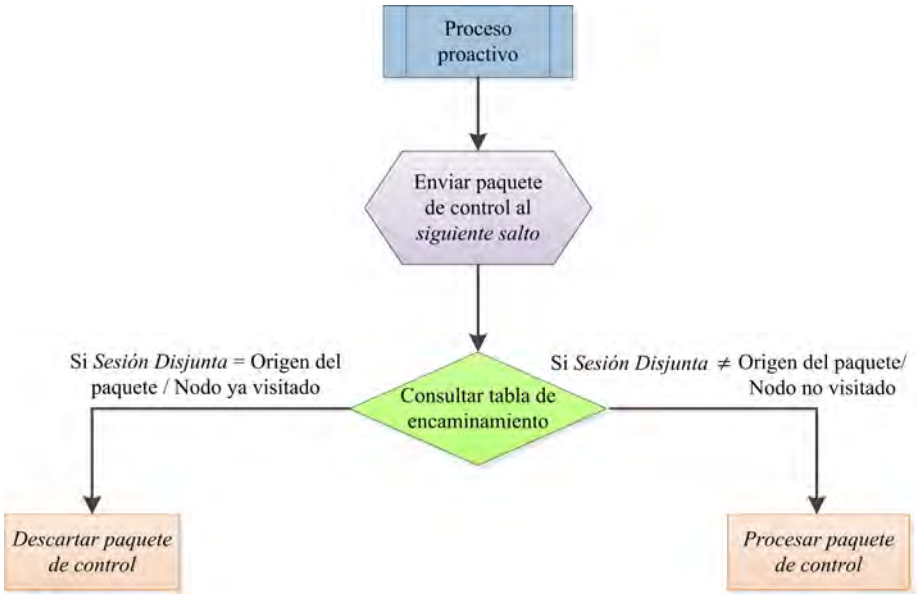


Figura 14.26: Diagrama de flujo de rutas de nodo disjunto (AntOR-DNR)

El Algoritmo 14.2 representa el pseudocódigo del proceso de cálculo de rutas de nodo disjunto.

```
mientras Proceso proactivo hacer
    {nexthop} = ObtenerSiguienteSalto(dst);
    Enviar(nexthop,msg);
    {src,dst} = ObtenerInfoSesion(msg);
    si ComprobarDisjuntoNodo(src) = CIERTO entonces
        Descartar(msg);
    fin
fin
```

Algoritmo 14.2: Cálculo de rutas de nodo disjunto (AntOR-DNR)

La línea 1 muestra un proceso proactivo que está representado con un bucle para indicar que se realiza continuamente después del inicio de la sesión de datos. En las líneas 2 y 3 se obtiene el siguiente salto (*nexthop*) y se envía la hormiga proactiva *hacia adelante* (paquete de control *msg*). En la línea 4 se obtiene el origen *src* y destino *dst* de la sesión de datos de este paquete de control. En la línea 5 se comprueba la propiedad disjunta de nodo, es decir, se verifica si ese origen *src* es el mismo que el del campo *Sesión Disjunta* de la tabla de encaminamiento. En caso afirmativo se cumple la línea 6 descartando dicho paquete por tratarse de una ruta disjunta de nodo.

14.6 AntOR - Restrictive Disjoint Link Route (AntOR-RDLR)

Como su nombre indica, *AntOR - Restrictive Disjoint Link Route* (AntOR-RDLR) se deriva del protocolo AntOR-DLR, presentando dos importantes diferencias respecto a éste. La primera, que da origen a su nombre, es la que ocurre en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas, donde, por un lado flexibiliza ésta al permitir a las hormigas proactivas la coexistencia de rutas no disjuntas de enlace con rutas disjuntas de enlace, y por otro, restringe éstas últimas a que contengan un número máximo de enlaces disjuntos. La segunda diferencia ocurre en la fase de establecimiento de ruta y está relacionada con el proceso de actualización de feromona. Seguidamente se profundiza en cada una de estas diferencias.

El proceso de actualización de feromona en la fase de establecimiento de ruta es como sigue:

Si el nodo i que tiene una ruta al destino d ya tiene un valor de feromona virtual y en la fase de establecimiento de ruta le llega otro de feromona regular aplicando la ecuación 14.7, entonces el valor de feromona regular sustituye a la feromona virtual usando el máximo de estos dos valores, y quedando a 0 el valor de feromona virtual. La ecuación 14.17 resume el proceso:

$$\begin{aligned} Regular_{last} &= F(Regular_{new}, time) \\ Regular_{final} &= max(Regular_{last}, Virtual_{old}) \\ Virtual_{final} &= 0 \end{aligned} \tag{14.17}$$

Como se ha comentado anteriormente, en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas de AntOR-DLR las hormigas hormigas proactivas *hacia adelante* no van por rutas disjuntas de enlace. Por el contrario, en AntOR-RDLR sí se permite elegir enlaces disjuntos para la retransmisión de los datos hasta un máximo de MAX_HOP saltos.

El Algoritmo 14.3 representa el proceso de cálculo de rutas en AntOR-RDLR.

```
mientras Proceso proactivo hacer
    {src,dst} = ObtenerInfoSesion(msg);
    {hop} = ObtenerSaltoPermitido(msg);
    {nexthop} = GetSiguienteSalto(dst);
    {link} = ObtenerEnlace(dst,nexthop);
    si ComprobarDisjuntoEnlace(link) = FALSO entonces
        Enviar(nexthop,msg);
    de lo contrario
        si hop ≤ MAX_HOP entonces
            hop = hop + 1;
            ActulizarSaltoPermitido(hop);
            Enviar(nexthop,msg);
        fin
    fin
fin
```

Algoritmo 14.3: Cálculo de rutas (AntOR-RDLR)

La línea 1 indica que continuamente se está produciendo un proceso proactivo. La línea 2 obtiene el origen *src* y destino *dst* de la sesión de datos con el método *ObtenerInfoSesion*. La línea 3 constituye el núcleo de este algoritmo. En esta línea se obtiene el contador actual *hop* de saltos permitidos, accediendo a un campo del mensaje msg que contiene información del número de nodos disjuntos que han sido recorridos por las hormigas proactivas. Las líneas 4 y 5 calculan el posible siguiente salto *nexthop* para encaminar el mensaje *msg* así como el posible enlace link. En la línea 6 se comprueba si el enlace es disjunto o no. Si no es disjunto (línea 7) se envía la correspondiente hormiga proactiva al siguiente salto nexthop. En caso afirmativo, esto es, si se trata de un enlace disjunto, se aplica la propiedad denominada restrictiva, que consiste en transmitir por este enlace disjunto (no permitido en AntOR-DLR) hasta un número máximo de veces MAX_HOP (línea 9). Las líneas 10 y 11 actualizan el valor del contador actual de saltos hop. Finalmente, se permite enviar (línea 12) la hormiga proactiva usando el siguiente salto *nexthop* previamente calculado.

En AntOR-DLR un enlace seleccionado de una ruta disjunta de enlace no es candidato para el envío en el proceso de retransmisión de los agentes proactivos. En AntOR-RDLR se puede retransmitir por dicho enlace hasta un número máximo de intentos MAX_HOP. Esto es posible gracias a un campo *Reserved* de la cabecera del paquete. En AntOR-DLR este campo tiene diferente función: número de saltos permitidos en modo *broadcast* en el proceso de reparación local de ruta. En AntOR-RDLR se usa para indicar el número actual de intentos de selección de una ruta disjunta en el proceso de exploración de nuevos caminos.

La Figura 14.27 muestra un ejemplo de la propiedad restrictiva.

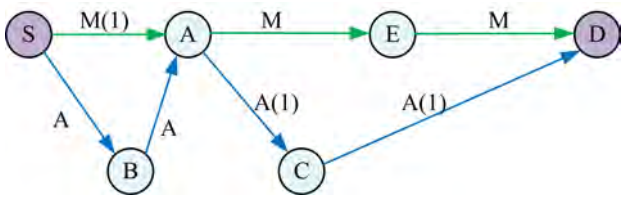


Figura 14.27: Ejemplo en el proceso de proactivo (AntOR-RDLR)

Se indica en color verde la ruta principal creada (etiqueta M) y en color azul la posible ruta alternativa (etiqueta A), que las hormigas proactivas *hacia adelante* (PFA) pueden explorar. Se muestra el contador cont actual de intentos como un número entre paréntesis después de la etiqueta. En este ejemplo el nodo S inicia un proceso proactivo de exploración de nueva ruta, enviando la correspondiente PFA al destino D. En AntOR-DLR la hormiga proactiva no puede ir por enlaces que pertenecen a la ruta principal (al tratarse de rutas disjuntas de enlace). En AntOR-RDLR se permite un número máximo (MAX_HOP) de posibilidades de elegir un enlace perteneciente a la ruta principal. Se establece MAX_HOP a un valor de 2 saltos permitidos. Finalmente, el nodo S selecciona el nodo A para retransmitir la PFA. Esto está permitido porque el número de intentos MAX_HOP se ha establecido a 2 y el actual contador cont sólo ha empleado 1, con lo cual no ha superado lo establecido. El siguiente nodo que se elige es C. Como el enlace formado por los nodos A-C no pertenece a la ruta principal, el contador actual cont no se incrementa. Lo mismo ocurre para el enlace formado por los nodos C-D. Cuando llega esta hormiga al destino, al igual que en AntOR-DLR, se envía la correspondiente hormiga proactiva *hacia atrás* (PBA) para actualizar las entradas de la tabla de encaminamiento de la ruta que ha sido indicada en la fase *hacia adelante*.

14.7 AntOR - Unicast Disjoint Link Route (AntOR-UDLR)

Como su nombre indica, *AntOR - Unicast Disjoint Link Route* (AntOR-UDLR) se deriva del protocolo AntOR-DLR, diferenciándose de éste en la fase de gestión de fallos de enlace.

Previo a la especificación de AntOR-UDLR conviene reseñar las diferencias de los mensajes *unicast* frente a los *broadcast*. Se entiende por *unicast* el envío de información desde un único emisor a un único receptor. Se entiende por *broadcast* el envío desde un único emisor a toda la red de forma indiscriminada. La modalidad *unicast* presenta la ventaja de que produce menos colisiones (y, consecuentemente, menos pérdidas de mensajes) pero conlleva un retardo adicional ya que comprueba por medio de mensajes de control que el canal está libre para transmitir.

AntOR-UDLR sustituye los mensajes de notificación enviados en modo *broadcast* en AntOR-DLR por mensajes (*unicast*) sencillos enviados al precursor que tiene una ruta válida a un destino alcanzable, entendiendo por ruta válida la perteneciente a la sesión activa de un destino dado con un valor positivo de feromona regular. Cuando un nodo detecta el fallo de enlace en su vecino, comunica a su antecesor dicho fallo por medio de un mensaje *unicast*, repitiéndose este proceso hasta llegar al nodo origen de la sesión de datos. Esto provoca que el origen lance una hormiga *hacia adelante* de establecimiento de ruta. Puede ocurrir que el nodo que detecta el fallo tenga dos o más sesiones de datos solapadas,

comunicándose éste a los nodos origen de las distintas sesiones de datos involucradas.

La Figura 14.28 ilustra el proceso de notificación de fallo de enlace en AntOR-UDLR:

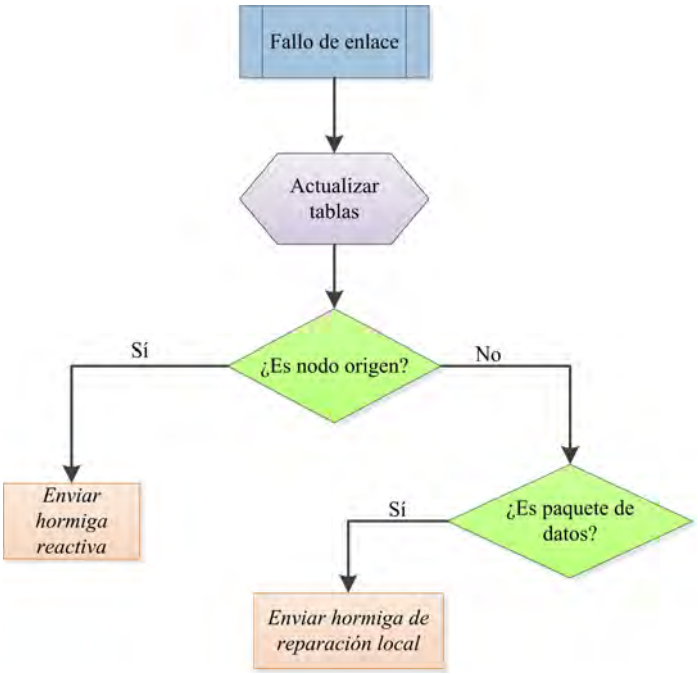


Figura 14.28: Gestión de Fallo de Enlace (AntOR-UDLR)

Cuando hay fallo de nodo, se produce fallo tanto de nodo como de enlace. El nodo que percibe el fallo elimina de su tabla de vecinos al correspondiente nodo. A continuación, actualiza la tabla de encaminamiento con la nueva información de feromona y procede como sigue:

a) Si no hay ruta en el origen, se envía una hormiga reactiva *hacia adelante*.

b) Si no hay ruta en un nodo intermedio y era un paquete de datos lo que se estaba retransmitiendo cuando se produjo el fallo, se envía una hormiga hacia adelante de reparación de ruta. Si no hay respuesta de la correspondiente hormiga *hacia atrás* de reparación en un determinado período de tiempo, se envía un mensaje *unicast* al precursor de la ruta informando de que el destino es inalcanzable. El nodo que recibe este mensaje actualiza la tabla de encaminamiento y reenvía este mensaje al precursor y así sucesivamente hasta llegar al nodo origen de la sesión de datos.

c) Si no hay ruta en el nodo intermedio y si se trata de un paquete de control (un mensaje *Hello* o un mensaje de control *unicast*), no se envía ningún mensaje. Esto puede originar que haya rutas que no se hayan podido reparar correctamente. Cuando un nodo intermedio que encamina los datos no encuentra una ruta válida envía un mensaje *unicast* a todos los vecinos a un salto para que actualicen sus tablas de encaminamiento. Es necesario enviar este mensaje a todos los vecinos porque, de lo contrario, al no encontrar ruta válida no hay información del predecesor. Cuando uno de estos nodos vecinos tiene una ruta válida al destino, reenvía el mensaje *unicast* al precursor de la ruta, y así sucesivamente hasta llegar al nodo origen.

La Figura 14.29 muestra la estructura del mensaje *unicast* de notificación de enlace, abreviadamente, *Unicast Link Notification* (ULN).

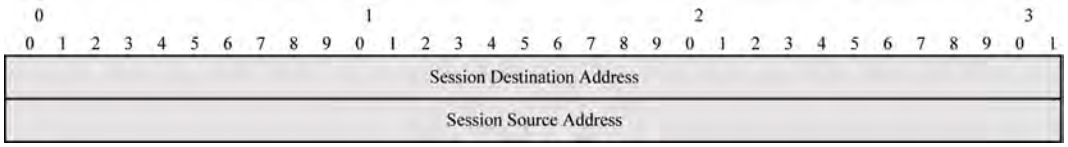


Figura 14.29: Formato del mensaje *unicast* de notificación de enlace (AntOR-UDLR)

Como se puede observar, tiene una estructura sencilla, conteniendo únicamente dos direcciones IP: *Session Destination Address* y *Session Source Address*. La primera dirección hace referencia al destino de la sesión de datos con ruta válida y la segunda hace referencia al origen. Se usa la dirección de destino porque, cuando se produce un fallo de enlace, el nodo que lo detecta debe indicar el destino para que los nodos predecesores puedan procesar el mensaje adecuadamente y decidir si lo reenvían en el supuesto de que haya ruta válida al destino. La dirección origen es también necesaria porque indica al nodo que recibe el mensaje si se ha alcanzado el origen, comprobando si la dirección origen encapsulada en el mensaje es la misma que la dirección principal del nodo.

El Algoritmo 14.4 muestra el pseudocódigo del proceso de neutralización de fallo de enlace.

```
{src,dst} = ObtenerInfoSesion(msg);
si ComprobarRutaValida(dst) entonces
    si NODO_ACTUAL = src entonces
        EnviarRFA();
    de lo contrario
        TTL = TTL - 1;
        {pre} = ObtenerPrecursor(dst);
        Reenviar(pre,msg);
    fin
fin
```

Algoritmo 14.4: Neutralización de fallo de enlace (AntOR-UDLR)

La línea 1 obtiene origen y destino de la sesión de datos. Esta información se extrae del mensaje ULN *msg*. La línea 2 comprueba si existe una ruta válida al destino *dst* (sesión activa y valor positivo de feromona regular). En caso afirmativo se comprueba (línea 3) si el nodo actual (el que recibe el mensaje *msg*) es equivalente a *src*. Si se ha alcanzado el origen de la sesión de datos se procesa un nuevo establecimiento de ruta (línea 4). En caso contrario (línea 5) se reenvía el mensaje *msg* (líneas 6 a 8). La línea 6 decrementa una unidad el valor del campo TTL. Este campo se incluye en la cabecera del paquete. La línea 7 consigue el precursor *pre*, para que en la línea 8 pueda hacerle el reenvío del mensaje *msg*.

La Figura 14.30 ilustra un ejemplo que explica el modo en el que se trata un fallo de enlace en un nodo intermedio cuando se transmite un mensaje de datos y no se consigue reparar la ruta (caso b). La red del ejemplo está formada por 5 nodos, siendo el nodo origen y el nodo destino A y E, respectivamente. Se marca en color rojo el nodo que falla, originando un fallo de enlace entre C y D. C notifica a su antecesor B con un mensaje *unicast* sencillo (ULN) que el destino E es inalcanzable. Al recibir B este mensaje lo reenvía a su precursor A, decrementando en una unidad el valor TTL de dicho mensaje. Finalmente, cuando el nodo origen A recibe este mensaje se ejecuta un nuevo proceso de establecimiento de ruta.

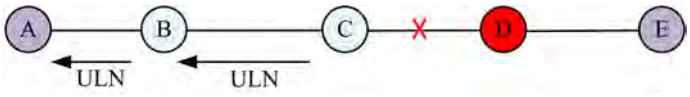
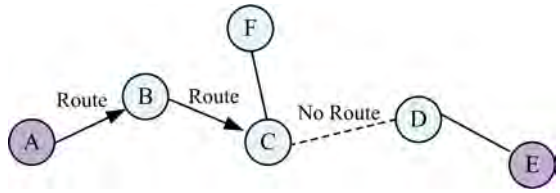
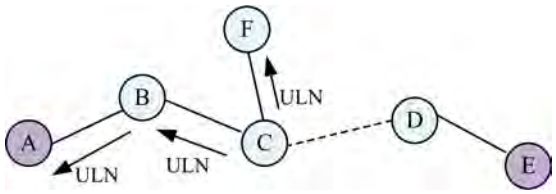


Figura 14.30: Ejemplo de gestión de fallo de enlace - caso b (AntOR-UDLR)

La Figura 14.31 ilustra el caso c comentado anteriormente que es específico de AntOR-UDLR. Esta figura está formada por 6 nodos, siendo A y E el origen y destino de una sesión de datos, respectivamente. Según la Figura 14.31(a) el nodo A reenvía el paquete de datos (Route) al destino alcanzable E a través del siguiente salto B. Al recibir correctamente el paquete de datos el nodo B lo reenvía al nodo C con destino E. Como ahora C no encuentra la ruta (No Route) al siguiente salto D, no lo puede retransmitir, por lo que la información no puede ser encaminada con éxito al destino. En este instante se aplica el proceso específico de AntOR-ULDR (véase Figura 14.31(b)) enviando un mensaje *unicast* (ULN) a los vecinos. Para poder enviar a los vecinos el correspondiente mensaje es necesario buscar las direcciones IP de cada uno de ellos en la tabla de vecinos, enviándose un mensaje *unicast* por cada dirección IP del vecino encontrado. Los nodos F y B reciben el mensaje enviado por C, pero D no porque está eliminado de la tabla de vecinos de C, ya que fue el que originó el fallo. El nodo B lo reenvía a A, puesto que pertenece a la sesión de datos (A, B, C, D, E). Al recibir A este mensaje envía una hormiga reactiva *hacia adelante* para proceder a un nuevo establecimiento de ruta. En cambio, el nodo F procesa el mensaje, pero no lo reenvía porque no pertenecía a la ruta válida al destino E.



(a)



(b)

Figura 14.31: Ejemplo de gestión de fallo de enlace - caso c (AntOR-UDLR)

14.8 AntOR-v2

Como su nombre indica, AntORv2 se deriva del protocolo AntOR (más concretamente de AntOR-DLR), si bien presenta importantes diferencias respecto a éste, a saber: *buffering* de paquetes de control, gestión de rutas obsoletas, gestión de fallos de envío y eliminación de la feromona virtual en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas. Seguidamente se analizan en detalle estas diferencias.

El *buffering* de los paquetes de control consiste en que éstos se almacenan para su posterior envío a sus correspondientes destinos cada cierto intervalo de tiempo. Este hecho

permite que haya sincronismo en el envío de los paquetes y que no se congestione la red, disminuyendo su colisión. Cada entrada en el buffer incluye la siguiente información: a) Socket por el cual se envía el paquete; b) el paquete de control o mensaje particular del protocolo; y, c) la dirección destino (puede ser una dirección *broadcast* o una dirección *unicast* enviada a un determinado nodo).

La gestión de rutas obsoletas reemplaza el proceso de evaporación de la feromona. Este evento se realiza cada cierto intervalo de tiempo y es como sigue:

- Cada entrada de la tabla de encaminamiento tiene un campo (*timestamp*) que indica cuando se creó o se actualizó por última vez dicha entrada.
- Si el campo *timestamp* asociado a cada ruta de la tabla de encaminamiento es menor que la diferencia entre el tiempo actual y un tiempo límite dado, se elimina de forma local (cada nodo) la citada entrada.
- El valor de este límite es importante. Un valor bajo hace que las rutas converjan lentamente, eliminándose rutas a destinos activos. Por el contrario, un valor alto implica una alta convergencia en la creación de las rutas con la consiguiente posibilidad de mantener rutas obsoletas.

La gestión de fallos de envío está relacionada con la tolerancia a fallos. Cuando se detecta un fallo se lanza un proceso de neutralización. En entornos altamente dinámicos (con más roturas de enlaces) el número de procesos de neutralización como reparación local de ruta es mayor, originando una importante sobrecarga. El mecanismo introducido pretende aliviar este hecho mediante la comprobación de la existencia de ruta válida (valor positivo de feromona regular) al vecino al que se va a transmitir. Sólo en el caso de que el camino exista, se envía el paquete de control.

La cuarta y última diferencia y, quizás la más significativa, es, como se ha comentado anteriormente, la eliminación de la feromona virtual en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas. Se pretende reducir la sobrecarga utilizando agentes proactivos que no necesiten rutas de feromona virtual. Estos agentes crean rutas alternativas que van de vecino a vecino hasta alcanzar el nodo destino. A la hora de seleccionar el siguiente salto, los agentes tienen en cuenta el valor máximo de feromona regular de los vecinos de 1 salto. De esta forma se alcanzan rutas alternativas, que además son disjuntas de enlace. Estas hormigas proactivas se envían cuando el número de rutas alternativas es menor de un cierto umbral.

La Figura 14.32 muestra un ejemplo de selección del siguiente salto en el proceso proactivo exploratorio. La ruta principal (A, B, E) de color rojo se crea en la fase de establecimiento de ruta. En la fase de exploración el nodo A envía la correspondiente PFA, teniendo que elegir entre sus 3 vecinos intermedios: B, C, D. Estos vecinos tienen valores de feromona regular de 20, 5 y 15, respectivamente. Estos valores de feromona son inversamente proporcionales a la estimación de tiempo generada por la recepción de los mensajes *Hello*. B es el mejor candidato para reenviar (mayor valor de feromona), pero pertenece a la ruta principal, por lo que se eligen entonces al siguiente mejor (en este caso el nodo intermedio D). Este proceso continúa a través de los nodos intermedios hasta alcanzar el nodo destino. Por último, cabe reseñar que la variable MAX_TTL (*Time To Live* máximo) de las PFAs controla el número máximo de saltos en las rutas alternativas.

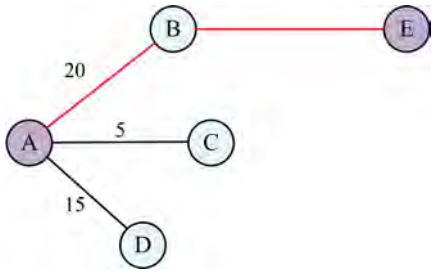


Figura 14.32: Ejemplo de proceso proactivo (AntOR-v2)

14.9 Hybrid ACO Routing (HACOR)

HACOR consiste en un refinamiento de AntOR-v2, diferenciándose de éste en la incorporación de la capacidad de buffering para los paquetes de datos, en un proceso optimizado de neutralización de fallos de enlace y en la introducción de un tipo particular de S-ACO en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas. Seguidamente se analizan en detalle estas diferencias.

El *buffering* de los paquetes de datos consiste en que éstos se almacenan para su posterior envío a sus correspondientes destinos cada cierto intervalo de tiempo en el supuesto de que no existan rutas. En efecto, cuando el paquete de datos está listo para enviarse al siguiente salto, comprueba si hay una ruta válida al destino perteneciente a la actual sesión de datos. En el caso de que no haya una ruta válida, se almacena el paquete de datos en la cola de paquetes, enviándose una hormiga *hacia adelante* de reparación local de ruta para solucionar el problema. Al mismo tiempo que se intenta reparar la ruta, el nodo envía un mensaje *unicast* a todos los vecinos alcanzables. Los vecinos que reciben este mensaje lo envían a sus precursores. En caso contrario, esto es, si hay una ruta válida, se procede al envío.

El Algoritmo 14.5 muestra el pseudocódigo del proceso de neutralización de fallos de enlace.

si *ComprobarEnlace() = CIERTO* ***y*** *ErrorTransmision() = CIERTO* **entonces**
  ActualizarVecinos() ;
  EliminarTodasRutas() ;
  si *ComprobarOrigen() = CIERTO* **entonces** EnviarRFA();
  de lo contrario si *ComprobarDato() = CIERTO* **entonces**
  EnviarHormigaReparacion();
  de lo contrario si *ComprobarHello() = FALSO* **entonces**
  EnviarUnicastPrecursor( );
fin

Algoritmo 14.5: Gestión de fallos de enlace (HACOR)

El primer evento que se produce cuando hay un fallo de nodo es que el nodo que lo percibe actualiza su tabla de vecinos, eliminando todas las rutas que tiene el nodo que falla como siguiente salto. Si no hay una ruta en el nodo origen se inicia el establecimiento de ruta enviando una hormiga reactiva *hacia adelante*. Si no hay ruta en un nodo intermedio y un paquete de datos se estaba reenviando cuando se produjo el fallo, se envía una hormiga *hacia adelante* de reparación local de ruta a cada uno de los destinos de todas las sesiones de datos afectadas. Si no hay ruta en el nodo intermedio y se estaba enviando un paquete de control (*Hello*) en modo *broadcast*, no se realiza ningún proceso de neutralización.

Si lo que se estaba reenviando era un paquete de control *unicast* se envía un mensaje ULN al nodo precursor. Este proceso se repite sucesivamente hasta alcanzar el nodo origen.

Por último, la tercera característica diferenciadora de HACOR respecto a su predecesor es la introducción de una variante de S-ACO en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas que consiste básicamente en lo siguiente:

a) La feromona virtual deja de ser necesaria en esta fase.

b) No se usa el proceso de evaporación.

c) Se usa un método libre de bucles (véase Figura 14.33) cuando la hormiga proactiva *hacia adelante* (PFA) ha llegado al nodo destino. Posteriormente, se elimina el bucle, convirtiéndose esta PFA en una PBA libre de bucles, que vuelve al origen por los nodos visitados de la lista, actualizando las tablas de encaminamiento de cada nodo.

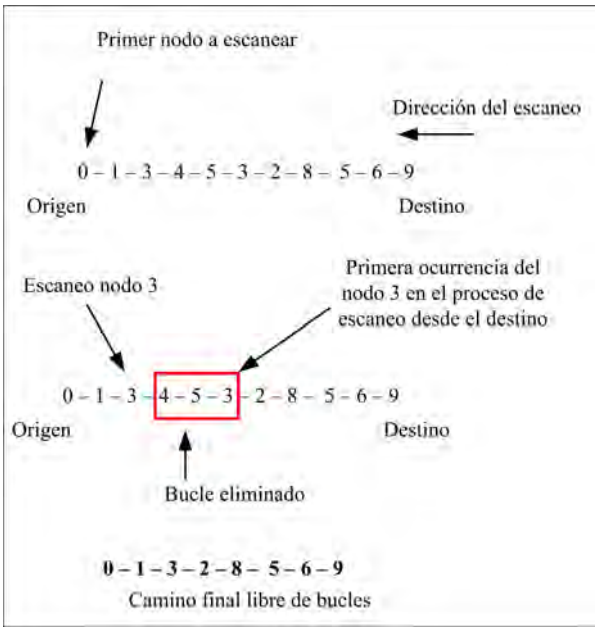


Figura 14.33: Proceso eliminación de bucles (HACOR)

d) No es necesario el establecimiento inicial de valores de feromona a cada vecino a 1 salto. El proceso de exploración se realiza salto a salto con la información de feromona que tienen los vecinos a un salto mediante la utilización de los mensajes *Hello*. Cada nodo que recibe un mensaje *Hello* de otro vecino a un salto actualiza su ruta con el nuevo valor de feromona.

e) Las hormigas proactivas *hacia adelante* utilizan rutas disjuntas de enlace. La Figura 14.34 presenta un esquema de este proceso exploratorio desde el nodo origen al nodo destino.

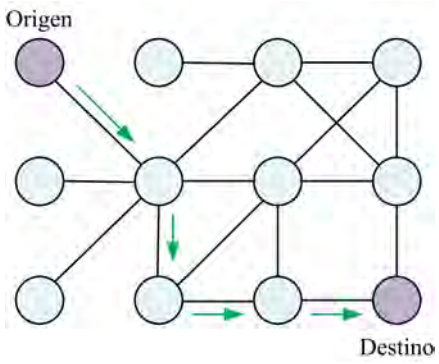


Figura 14.34: Ejemplo de exploración de caminos (HACOR)

f) Esta utilización de rutas disjuntas implica la comprobación de si el vecino a un salto que tiene que reenviar el correspondiente agente proactivo pertenece o no a una ruta disjunta. En el caso de que el vecino pertenezca a una ruta disjunta no se elige (con objeto de reducir la sobrecarga).

14.10 Parallel AntOR (PAntOR)

Como su nombre indica, AntOR Paralelo deriva de AntOR, más concretamente, puede considerarse una aproximación paralela de AntOR-DNR. La razón de elegir AntOR-DNR (y no AntOR-DLR) es que se pretende analizar el caso peor, de ahí la elección del primero por ser más restrictivo. Previo a la especificación de PAntOR conviene puntualizar algunos aspectos de la paralelización de los algoritmos ACO.

En primer lugar, conviene saber que la práctica totalidad de los trabajos relativos a algoritmos ACO paralelos están diseñados para sistemas centralizados, basados en técnicas maestro esclavo, donde el maestro central distribuye trabajo a los demás procesadores. PAntOR, por su parte, está diseñado para sistemas descentralizados, lo que le confiere aún mayor relevancia.

En segundo lugar, conviene saber que los algoritmos ACO paralelos se clasifican según los dos criterios que se describen a continuación: Una posible clasificación diferencia si el algoritmo de una paralelización de ACO es estándar o es especialmente diseñado. El objetivo de una paralelización ACO estándar es disminuir el tiempo de ejecución sin cambiar el comportamiento del algoritmo. Por el contrario, los algoritmos paralelos específicos cambian ACO en aras de obtener un algoritmo más eficiente. Un método empleado para diferenciar estas dos aproximaciones consiste en cómo se hace uso del intercambio de información entre los procesadores.

Otra posible clasificación comprueba si el algoritmo tiene un enfoque centralizado o descentralizado. En un enfoque centralizado es normal que sea un procesador el que recopila la información de feromonas, así como las diferentes soluciones de los demás procesadores. De esta forma la actualización de la feromona se hace de forma centralizada. En un enfoque descentralizado cada procesador tiene que calcular la actualización de la feromona por sí mismo utilizando la información que ha recibido de otros procesadores.

PAntOR consiste en una paralelización ACO estándar (paralelización de grano grueso) con un enfoque descentralizado.

Para entender el funcionamiento de PAntOR conviene introducir previamente tres conceptos:

a) *Proceso*: programa en ejecución. Los procesos son gestionados por el sistema operativo.

b) *Hilo*: unidad básica de ejecución. Cualquier programa que se ejecuta consta de al menos un hilo.

c) *Portable Operating System Interface (POSIX) Thread*: estándar basado en Application Programming Interface (API) de hilos para C/C++.

Se utiliza POSIX Thread porque permite expandir un nuevo flujo de procesos concurrentes, siendo lo más eficiente en sistemas multi-core donde el flujo de procesos puede programarse para ser ejecutado en otro procesador, ganando así velocidad a través del procesamiento distribuido o paralelo. La programación mediante hilos conlleva menos sobrecarga que el tener que expandir un nuevo proceso, porque el sistema no tiene que inicializar un nuevo entorno ni un espacio de memoria virtual para ese proceso.

Las tecnologías de programación paralela, tales como *Message Passing Interface* (MPI) y *Parallel Virtual Machine* (PVM), se usan en un entorno de computación distribuida, mientras que los hilos se limitan a un sistema de una sola computadora. Todos los hilos dentro de un proceso comparten el mismo espacio de direcciones. Para que la ejecución de este algoritmo de encaminamiento sea más rápida se utilizan las librerías de POSIX Thread. Esta técnica paralela consiste en lanzar un hilo por cada vecino que inicia alguno de los siguientes procesos: establecimiento de ruta, reparación local de ruta y notificación de fallo de enlace.

La Figura 14.35 muestra el diagrama de flujo del paralelismo introducido en el proceso de establecimiento de ruta. Este proceso se paraleliza por medio de hilos, enviándose una hormiga reactiva *hacia adelante* a los vecinos que se encuentran a un salto mediante un hilo independiente, siendo el número de hilos que se utilizan proporcional al número de vecinos del nodo que inicia este proceso. Cuando un nodo intermedio recibe esta hormiga repite el proceso. En cambio, si se trata de un nodo destino, éste envía su correspondiente hormiga reactiva *hacia atrás* (RBA).

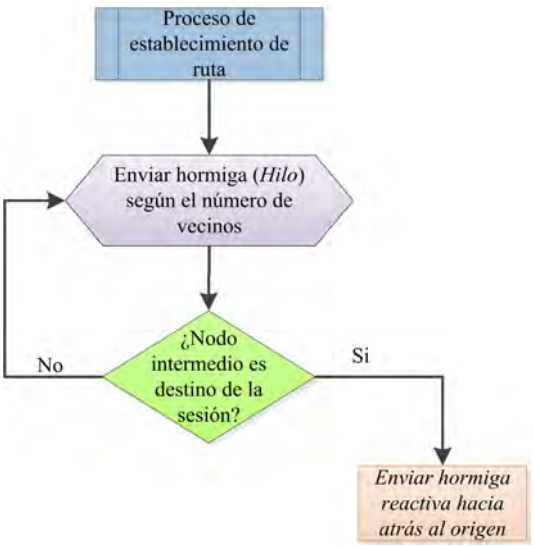


Figura 14.35: Paralelización del proceso de establecimiento de ruta (PAntOR)

La Figura 14.36 muestra el diagrama de flujo del paralelismo introducido en el proceso

de reparación local de ruta. Como puede observarse, su funcionamiento es análogo al comentado en el proceso establecimiento de ruta, salvo que se realiza a nivel local.

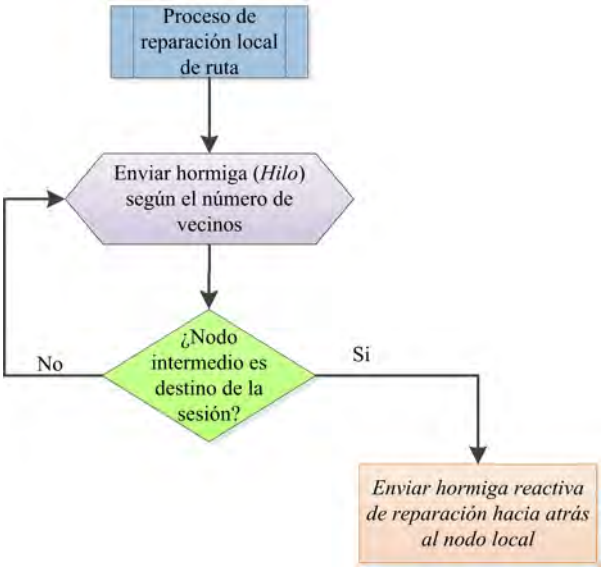


Figura 14.36: Paralelización del proceso de reparación local de ruta (PAntOR)

La Figura 14.37 muestra el diagrama de flujo del paralelismo introducido en el proceso de notificación de fallo de enlace. Como ya se ha comentado en el apartado 14.3.4, este proceso tiene como objetivo actualizar la tabla de encaminamiento ante los fallos de enlace. Es una fase de gran importancia, siendo crucial que sea realizada con rapidez. Los nodos envían hormigas en hilos independientes hasta que un nodo intermedio tenga alguna ruta alternativa al destino después de actualizar la tabla de encaminamiento.

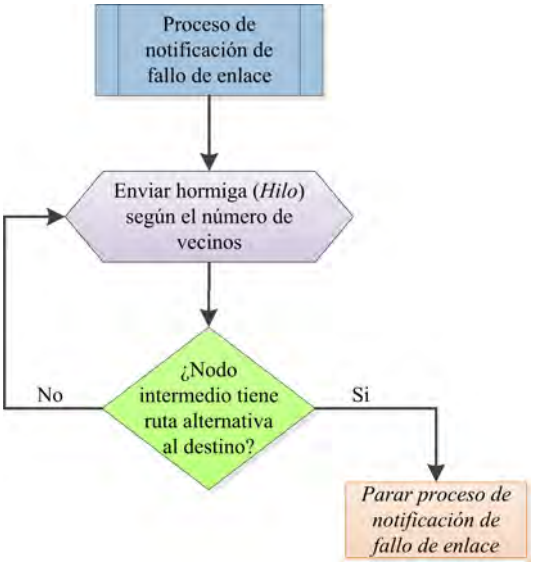


Figura 14.37: Paralelización del proceso de notificación de fallo de enlace (P-AntOR)

La Figura 14.38 muestra un ejemplo de funcionamiento de PAntOR. Si el nodo A

quiere iniciar el proceso de establecimiento de ruta en AntOR, consulta los candidatos a los que enviar una hormiga reactiva *hacia adelante* en su tabla de vecinos N = $\{N_1, N_2, N_3\}$ en un hilo independiente. En PAntOR se envían 3 hilos.

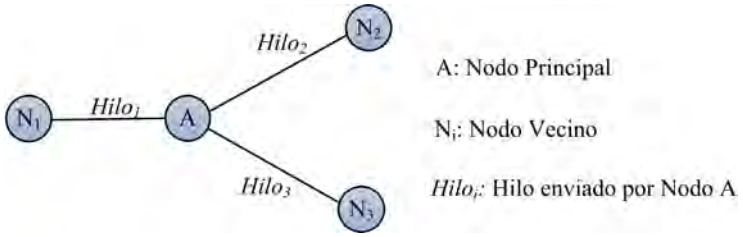


Figura 14.38: Ejemplo del funcionamiento (PAntOR)

14.11 PAntOR - Multiple Interface (PAntOR-MI)

Como su nombre indica, PAntOR-MI es una variante de PAntOR concebida para dispositivos que contengan más de una interfaz, esto es, para dispositivos pequeños y portables con más de una antena o interfaz de red inalámbrico (PocketPC, teléfonos móviles de última generación, etc.). PAntOR-MI paraleliza el envío de hormigas *broadcast* a través de las interfaces mediante hilos. Debido a la dificultad de encontrar hardware especializado, PAntOR-MI sólo se ha aplicado al proceso de establecimiento de ruta utilizando dos interfaces.

El Algoritmo 14.6 muestra el proceso de establecimiento de ruta en PAntOR-MI. Como puede observarse, mientras se ejecuta el proceso de establecimiento de ruta, se envía un mensaje reactivo en modo *broadcast* por cada interfaz que tenga el nodo, gestionándolo dicho interfaz por medio de un hilo.

```
mientras Proceso de establecimiento ruta hacer
    para Cont = 1 hasta Max_Interfaces hacer
        Enviar Mensaje Broadcast por Hilo(Cont);
    end
fin
```

Algoritmo 14.6: Establecimiento de ruta (PAntOR-MI)

La Figura 14.39 muestra un ejemplo del funcionamiento de PAntOR-MI. Se asume que el medio (canal inalámbrico) es el mismo para todos los dispositivos, y que todos los nodos son homogéneos (iguales capacidades computacionales y con idénticos rangos de transmisión).

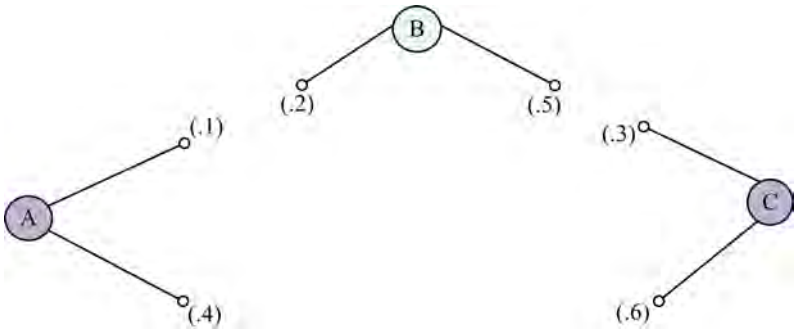


Figura 14.39: Funcionamiento PAntOR-MI

En este ejemplo se tienen 3 nodos (A, B y C) con dos interfaces de red cada uno. El nodo origen es A y el nodo destino es C. Se les asocia una única dirección IP a cada interfaz de red. Cada nodo considera dirección principal a una de estas direcciones IP. Las direcciones principales para los nodos A, B y C son, respectivamente, (.1), (.2) y (.3). Cada dirección de interfaz tiene asociada su correspondiente dirección principal IP y cada nodo almacena la siguiente información para sus dos interfaces: (*Dirección IP del Interfaz*, *Dirección IP Principal*). Por ejemplo, el nodo A almacena la siguiente información por interfaz: (.1, .1) y (.4, .1).

Los paquetes de datos son retransmitidos por las direcciones principales y la funcionalidad de PAntOR-MI consiste en gestionar por medio de un hilo el envío de las RFAs. Se envían en modo *broadcast* para no dividir las rutas creadas en el proceso de establecimiento de ruta y para asegurar que el mensaje RFA enviado por una antena (interfaz de salida) del nodo llega (si no es el caso, comprueba que se recibe por el otro interfaz).

Si cada interfaz del mismo nodo recibe una RFA, la descarta porque se trata del mismo nodo. En cambio, en los nodos intermedios no se descarta porque se comprueba que su dirección principal no corresponde a la de su interfaz.

Las RFA en su lista de nodos visitados van almacenando la dirección IP principal asociada a cada interfaz visitado. Al llegar la RFA al nodo destino C, se procesa la información de la RFA, convirtiéndose en una RBA que regresa salto a salto en modo *unicast* al nodo origen A, utilizando la información aprendida de las direcciones almacenadas en la lista de nodos visitados. En este proceso de vuelta se van actualizando o creando las rutas en los nodos intermedios con las direcciones principales. Esta aproximación sólo tiene en cuenta las direcciones principales porque los datos se encaminan por ellas y porque se pretende crear las rutas lo antes posible en el proceso de establecimiento.

14.12 Resumen

El principal objetivo de este capítulo ha sido la especificación de una familia de protocolos de encaminamiento ACO para redes móviles ad hoc. Estos protocolos tienen una raíz común: el protocolo AntOR, inspirado en AntHocNet, del que hereda su carácter híbrido, multicamino y adaptativo. Se ha comenzado viendo que la utilización de rutas disjuntas de nodo y/o de enlace, la separación entre la feromona regular y la virtual y algunos cambios introducidos en la fase de mantenimiento de rutas establecidas y exploración de nuevas rutas son las principales diferencias de AntOR respecto a su predecesor. Posteriormente, se han presentado las estructuras de datos del protocolo y se han descrito detalladamente las cuatro fases del mismo. Luego, se han mostrado los protocolos específicos que se de-

rivan del protocolo base AntOR, viendo en primer lugar las variantes AntOR disjunto de enlace (AntOR-DLR) y AntOR disjunto de nodo (AntOR-DNR) que utilizan rutas que no comparten enlaces/nodos, respectivamente. A continuación, se ha indicado cómo estas variantes dan lugar a otros protocolos por refinamientos sucesivos. Más concretamente, se ha visto cómo AntOR-RDLR, AntOR-UDLR, AntOR-v2 y HACOR derivan de AntOR-DLR. Así, AntOR-RDLR difiere de AntOR-DLR en el proceso de actualización de feromonas y en el mecanismo de exploración de rutas; AntOR-UDLR es una aproximación *unicast* de AntOR-DLR sustituyendo sus mensajes de notificación de fallo de enlace que se envían en modo difusión (*broadcast*) por mensajes *unicast* que se envían al predecesor del nodo que informa del fallo del enlace; AntOR-v2 incorpora el *buffering* de paquetes de control y la gestión de rutas obsoletas y de fallos de envío, eliminando el uso de la feromona virtual en la fase de exploración de nuevas rutas; y HACOR incorpora además el *buffering* de los paquetes de datos, una óptima neutralización de fallos y el uso de S-ACO en la exploración de rutas. Finalmente, también se ha visto cómo PAntOR y PAntOR-MI derivan de AntOR-DNR: PAntOR es una paralelización ACO estándar del protocolo AntOR-DNR con un enfoque descentralizado y PantOR-MI es una versión de PAntOR concebida para dispositivos multi-interfaz.

Capítulo 15

Simulaciones y Resultados

Este capítulo presenta las simulaciones realizadas utilizando diversos escenarios reales que comprueban la aplicabilidad de las diferentes propuestas. Para ello se ha utilizado el simulador de redes *Network Simulator 3* [NS3], uno de los más utilizados en el área. Primero se argumenta la elección del simulador de redes elegido. A continuación se describen los escenarios de simulación utilizados. Posteriormente, se comentan las métricas que se han analizado como, por ejemplo, *throughput*, ratio de paquetes entregados, retardo medio extremo a extremo, *jitter*, sobrecarga en el número de paquetes, sobrecarga en el número de bytes, etc. El capítulo finaliza con una breve síntesis de lo expuesto en el mismo.

15.1 Elección del Simulador de Redes

Las simulaciones se utilizan como apoyo en el diseño de protocolos. Hay dos aspectos importantes que deben evaluarse antes de la realización de las mismas: uso del modelo adecuado y elección de la mejor herramienta para el modelo en cuestión. A continuación se presentan los simuladores de redes más relevantes así como las características de cada uno de ellos:

- ***Network Simulator 2* (NS-2)**: *Network Simulator 2* (NS-2) [NS2] es un simulador de eventos discretos utilizado principalmente en ambientes académicos y de investigación. Las simulaciones se componen de código escrito en C++ (que es usado para modelar el comportamiento de los nodos simulados) y por secuencias de comandos *Object-Oriented Tool Command Language* (oTcl) (que controlan la simulación y especifican aspectos adicionales como la topología de la red). Este diseño fue elegido para evitar recompilaciones innecesarias cuando se hacían cambios en la estructura de la simulación ya que una frecuente recompilación del programa en C++ consumía mucho tiempo cuando salió la primera versión. Sin embargo, actualmente esto no es un problema y no es necesario sacrificar el rendimiento de la simulación para ahorrar en recompilaciones, sobre todo cuando se simula una red de gran tamaño [BHvR05].

- ***Network Simulator 3* (NS-3)**: Al igual que su predecesor, NS-3 [NS3] es un simulador de eventos discretos y se basa en C++ para la implementación de los modelos de la simulación. Sin embargo, NS-3 ya no utiliza secuencias de comandos oTcl para controlar la simulación evitando los problemas presentados por la combinación de C++ y oTcl en NS-2. Los escenarios de simulación en NS-3 pueden implementarse en C++ puro y, opcionalmente, partes de la simulación se pueden realizar utilizando Python.

- ***Objective Modular Network Testbed in C++* (OMNeT++)**: En contraste con NS-2 y NS-3, OMNeT++ [OMN] no es un simulador de red por definición, sino un simulador de propósito general basado en eventos discretos. Sin embargo, se aplica sobre todo al dominio de simulación de redes, teniendo en cuenta el hecho de que su paquete *Integrated Network Enhanced Telemetry* (INET) ofrece una amplia colección de modelos de protocolos de Internet. Las simulaciones consisten en los llamados módulos simples que realizan el comportamiento de un modelo, por ejemplo, un determinado protocolo. Se pueden unir varios módulos simples para formar un módulo compuesto [BHvR05]. Al igual que NS-2 y NS-3, OMNeT++ se basa en C++ para la implementación de los módulos simples. La composición de estos módulos simples en módulos compuestos y, por tanto, la configuración de la simulación, se lleva a cabo en *Network Description* (NED), lenguaje de descripción de red de OMNeT++.

De los simuladores de red mencionados es NS-2 el más utilizado en el ámbito académico y de investigación. Sin embargo, muchos de sus usuarios se quejan de la complejidad propia del simulador y del alto consumo de recursos que lleva a la falta de escalabilidad, impidiendo la ejecución de simulaciones de redes con cientos de nodos [Kök08]. Esto se debe a que el tiempo de simulación aumenta exponencialmente con el número de nodos de la red y además consume mucha memoria al ejecutar la simulación.

Debido a todos estos problemas se creó NS-3. Uno de sus principales objetivos fue eliminar el problema de escalabilidad y soportar la simulación de manera paralela y distribuida [HRFR06]. A pesar de que NS-3 no tiene todos los modelos que tiene actualmente NS-2, posee más detalles de los modelos del estándar IEEE 802.11 y es posible integrarle nuevos módulos posibilitando que el simulador se actualice, permitiéndole seguir el rápido crecimiento de las redes inalámbricas [HRFR06]. Adicionalmente, NS-3 tiene nuevas funcionalidades como son: manejo correcto de múltiples interfaces, uso de direcciones IP, genera archivos PCAP que se utilizan para el análisis, etc.

En cuanto a OMNeT++ es un simulador bien organizado, flexible y fácil de usar. Sin embargo, posee informes bastante pobres de los resultados de la simulación, por lo que los usuarios deben desarrollar el código para obtener las métricas deseadas. Tiene extensiones externas las cuales permiten proveer soporte para la simulación de redes inalámbricas. Sin embargo, sólo es posible simular algunos escenarios ya que estas extensiones no están completas, sobre todo la de movilidad, además de que la documentación todavía está en fase de desarrollo y el análisis de las métricas de rendimiento es deficiente.

En [WvLW09] se demuestra que OMNeT++ requiere más tiempo que NS-3 para realizar una simulación, mientras que NS-2 no escala bien y, por tanto, no es adecuado para simulaciones de redes a gran escala. Asimismo, NS-3 es el simulador más eficiente con respecto al uso de la memoria.

Las consideraciones anteriores determinaron la elección del simulador de redes NS-3.

15.2 Entorno de Simulación

Para la realización de las simulaciones de los protocolos de encaminamiento ACO se ha considerado el siguiente escenario genérico:

- Todos los nodos se configuran en la capa física aplicando el estándar IEEE 802.11b con un rango de transmisión de 300 metros.

- En la capa de aplicación se usa Constant Bit Rate (CBR) para generar el tráfico de cada sesión de datos.

- La distribución de los nodos es aleatoria.

- El patrón de movilidad utilizado es Random WayPoint (RWP). En este modelo los nodos se mueven a destinos según marca la aleatoriedad de este patrón RWP y, una vez alcanzado tal destino, los nodos se detienen según el tiempo de pausa establecido, para a continuación seleccionar otro destino al que moverse.

- Los experimentos realizados se agrupan en tres tipos: variación en el escenario del tiempo de pausa, del número de nodos y de la velocidad de los mismos.

Tabla 15.1: Parámetros AntOR-DLR

Parámetro	**Valor**
Número de nodos	[20 - 100] nodos.
Distribución de los nodos	Aleatoria.
Área de simulación	1400 m × 1400 m.
Tiempo de simulación	30 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	3
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	0 s.
Finalización del tiempo CBR *cliente*	30 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	30 s.
Número de sesiones de datos	4
Tasa de datos	2048 bits/s (4 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[0 - 10] m/s.
Tiempo de pausa	5 s.

Las Tablas 15.1 a 15.9 a 15.16 presentan los parámetros de los escenarios y de los protocolos, respectivamente, utilizados durante las simulaciones.

Tabla 15.2: Parámetros AntOR-DNR

Parámetro	**Valor**
Número de nodos	100 nodos.
Distribución de los nodos	Aleatoria.
Área de simulación	1000 m × 1000 m.
Tiempo de simulación	120 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	3
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	0 s.
Finalización del tiempo CBR *cliente*	120 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	120 s.
Número de sesiones de datos	5
Tasa de datos	2048 bits/s (4 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[0 - 10] m/s.
Tiempo de pausa	[0 - 120] s con intervalos de 30 s.

Tabla 15.3: Parámetros AntOR-RDLR

Parámetro	**Valor**
Número de nodos	100 nodos.
Distribución de los nodos	Aleatoria.
Área de simulación	1000 m × 1000 m.
Tiempo de simulación	120 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	3
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	0 s.
Finalización del tiempo CBR *cliente*	120 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	120 s.
Número de sesiones de datos	5
Tasa de datos	2048 bits/s (4 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[0 - 10] m/s.
Tiempo de pausa	[0 - 120] s con intervalos de 30 s.

Tabla 15.4: Parámetros AntOR-UDLR

Parámetro	**Valor**
Número de nodos	100 nodos.
Distribución de los nodos	Aleatoria.
Área de simulación	3000 m × 1000 m.
Tiempo de simulación	300 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	5
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	Distribución Uniforme [0 - 60] s.
Finalización del tiempo CBR *cliente*	300 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	300 s.
Número de sesiones de datos	10
Tasa de datos	512 bit/s (1 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[2 - 10] m/s con intervalos de 2 m/s.
Tiempo de pausa	[0 - 240] s con intervalos de 60 s.

Tabla 15.5: Parámetros AntOR-v2

Parámetro	**Valor**
Número de nodos	[50 - 150] nodos.
Distribución de los nodos	Aleatoria.
Tiempo de simulación	300 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	10
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	Distribución Uniforme [0 - 60] s.
Finalización del tiempo CBR *cliente*	300 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	300 s.
Número de sesiones de datos	10
Tasa de datos	512 bit/s (1 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[0 - 8] m/s.
Tiempo de pausa	[0 - 240] s con intervalos de 60 s.

Tabla 15.6: Parámetros HACOR

Parámetro	**Valor**
Número de nodos	[50 - 150] nodos.
Distribución de los nodos	Aleatoria.
Tiempo de simulación	900 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	10
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	Distribución Uniforme [0 - 180] s.
Finalización del tiempo CBR *cliente*	900 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	900 s.
Número de sesiones de datos	10
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	5 m/s
Tiempo de pausa	[0 - 240] s con intervalos de 60 s.

Tabla 15.7: Parámetros PAntOR

Parámetro	**Valor**
Número de nodos	100 nodos.
Distribución de los nodos	Aleatoria.
Área de simulación	1200 m × 1200 m.
Tiempo de simulación	120 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	3
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	0 s.
Finalización del tiempo CBR *cliente*	120 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	120 s.
Número de sesiones de datos	5
Tasa de datos	2048 bit/s (4 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[0 - 10] m/s con intervalos de 2,5 m/s.
Tiempo de pausa	[0 - 120] s con intervalos de 30 s.
Número de núcleos	4
Memoria RAM	4 GBytes
Sistema Paralelo	Hilos mediante estándar Posix Thread

Tabla 15.8: Parámetros PAntOR-MI

Parámetro	**Valor**
Número de nodos	100 nodos.
Distribución de los nodos	Aleatoria.
Área de simulación	1200 m × 1200 m.
Tiempo de simulación	120 s.
Capa Física	IEEE 802.11
Rango de transmisión	300 m
Número de ejecuciones	3
Generador de tráfico	*Constant Bit Rate* (CBR).
Comienzo del tiempo CBR *cliente*	0 s.
Finalización del tiempo CBR *cliente*	120 s.
Comienzo del tiempo CBR *servidor*	0 s.
Finalización del tiempo CBR *servidor*	120 s.
Número de sesiones de datos	5
Tasa de datos	2048 bit/s (4 paquetes de 64 Bytes por segundo).
Patrón de movilidad	*Random WayPoint* (RWP).
Velocidad de los nodos	[0 - 10] m/s con intervalos de 2,5 m/s.
Tiempo de pausa	2
Número de núcleos	4
Memoria RAM	4 GBytes
Sistema Paralelo	Hilos mediante estándar Posix Thread

Tabla 15.9: Características internas de AntOR-DLR

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Parámetro 6 β_3	2
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	5
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2

Tabla 15.10: Características internas de AntOR-DNR

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Parámetro 6 β_3	2
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	5
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2

Tabla 15.11: Características internas de AntOR-RDLR

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Parámetro 6 β_3	2
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	5
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2

Tabla 15.12: Características internas de AntOR-UDLR

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Parámetro 6 β_3	2
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	3
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2

Tabla 15.13: Características internas de AntOR-v2

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	3
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2
Tiempo límite en gestión de rutas obsoletas	5

Tabla 15.14: Características internas de HACOR

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	3
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2
Tiempo límite en gestión de rutas obsoletas	5

Tabla 15.15: Características internas de PAntOR

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Parámetro 6 β_3	2
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	5
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2

Tabla 15.16: Características internas de PAntOR-MI

Parámetro	**Valor**
Parámetro 1 γ	0,7
Parámetro 2 α	0,7
Parámetro 3 η	0,7
Parámetro 4 β_1	20
Parámetro 5 β_2	20
Parámetro 6 β_3	2
Número de destinos en mensaje HELLO	10
Intervalo de emisión de HELLO	1 s.
Intervalo de emisión de PFA	2 s.
Número de intentos para establecer ruta	5
Número de *broadcast* permitidos por RRFA	2
Número de HELLOs consecutivos que pueden perderse	2

15.3 Métricas de Rendimiento

Las métricas de rendimiento para evaluar los protocolos de encaminamiento se dividen en métricas de efectividad y métricas de eficiencia. Las métricas de efectividad se consideran medidas externas al protocolo, porque miden si su rendimiento es el esperado a la hora de ejecutar la tarea para la cual fue diseñado. Como medidas de efectividad se distinguen: volumen de trabajo (*throughtput*), ratio de entrega de paquetes de datos, retardo medio extremo a extremo y *jitter*. Por otro lado, las métricas de eficiencia se consideran internas y se refieren a la sobrecarga generada. Destacan la sobrecarga en el número de paquetes y la sobrecarga en el número de bytes.

Las métricas de rendimiento definidas para la evaluación de los protocolos de encaminamiento ACO diseñados han sido las siguientes:

- ***Throughput***: Volumen de información que fluye a través de un sistema. Se calcula dividiendo el total de bits entregados al destino por el tiempo de entrega de paquetes.
- **Ratio de Entrega de Paquetes de Datos**: Relación entre el número de paquetes de datos entregados correctamente al destino y el número total de paquetes enviados.
- **Retardo Medio Extremo a Extremo**: Tiempo promedio de la transmisión de un paquete de datos por la red desde el origen al destino.
- ***Jitter***: Medida de la variación del tiempo de llegada de paquetes de datos consecutivos. Esta métrica, clasificada de robustez y adaptabilidad, es importante en las aplicaciones de QoS.
- **Sobrecarga en el Número de Paquetes**: Relación entre el número de paquetes de control enviados y el número de paquetes de datos correctamente entregados.
- **Sobrecarga en el Número de Bytes**: Relación entre el número total de bytes enviados y el número de bytes de los paquetes de datos entregados correctamente.

15.4 Evaluación del Protocolo AntOR-DLR

Para evaluar las prestaciones del protocolo AntOR-DLR, tanto en términos de eficiencia como de efectividad, se ha tenido en cuenta el impacto del incremento del número de nodos en la red (se ha utilizado el misma área de simulación variando la densidad de los nodos), esto es, cómo afecta éste a parámetros como el *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, la sobrecarga en el número de paquetes y la sobrecarga en el número de bytes. Esta evaluación se ha realizado conjuntamente con la del protocolo AntHocNet.

15.4.1 Throughput

Como se observa en la Figura 15.1, el *throughput* en AntOR crece de forma ligeramente lineal con el número de nodos. Asimismo, se observa cómo en redes densas supera ampliamente a AntHocNet, siendo además mucho más estable que éste. La razón de esta mejora en el throughtput se debe a que al aumentar el número de nodos se aumenta también el número de rutas alternativas fiables. Todo lo anterior permite inferir la escalabilidad del protocolo.

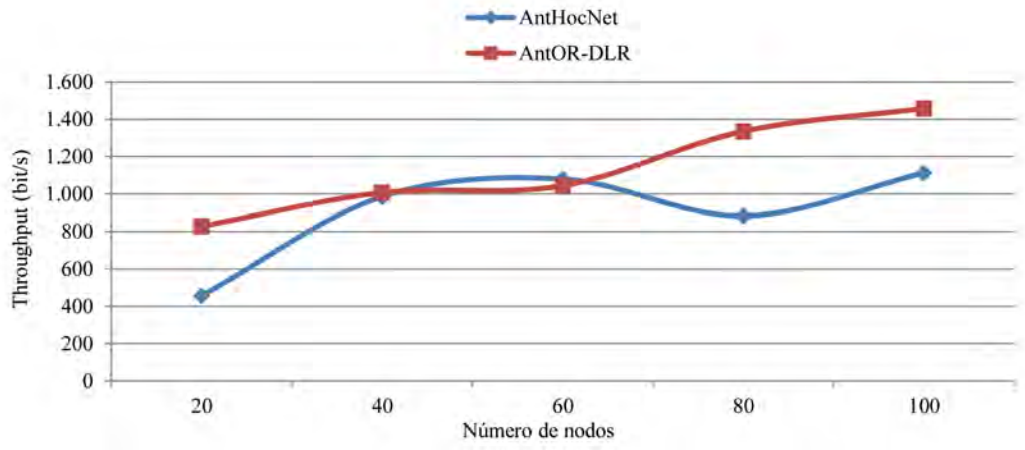


Figura 15.1: Throughput (AntOR-DLR)

15.4.2 Ratio de Entrega de Paquetes de Datos

Como se observa en la Figura 15.2, el ratio de entrega de paquetes de datos en AntOR tiene un comportamiento análogo al de *throughput*.

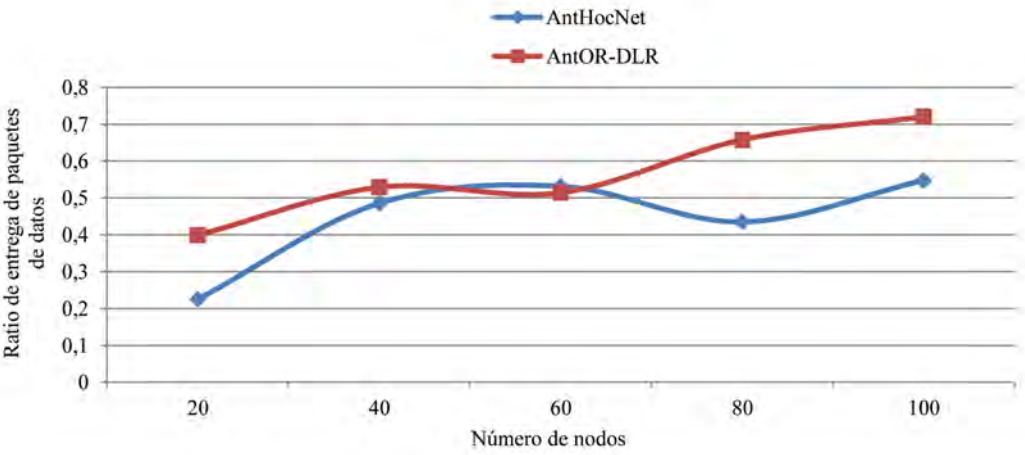


Figura 15.2: Ratio de entrega de paquetes de datos (AntOR-DLR)

15.4.3 Retardo Medio Extremo a Extremo

Como se observa en la Figura 15.3, el retardo medio extremo a extremo en AntOR es superior al de AntHocNet, siendo esta diferencia casi imperceptible (la escala está en milisegundos) en redes densas. Asimismo, otro aspecto reseñable es que, en términos generales, el retardo medio extremo a extremo en AntOR es más estable que el correspondiente a AntHocNet. Esta diferencia en el retardo medio extremo a extremo (que se reduce conforme aumenta el número de nodos) es consecuencia del hecho de que el mecanismo disjunto de enlace necesita de un mínimo número de nodos para ser efectivo.

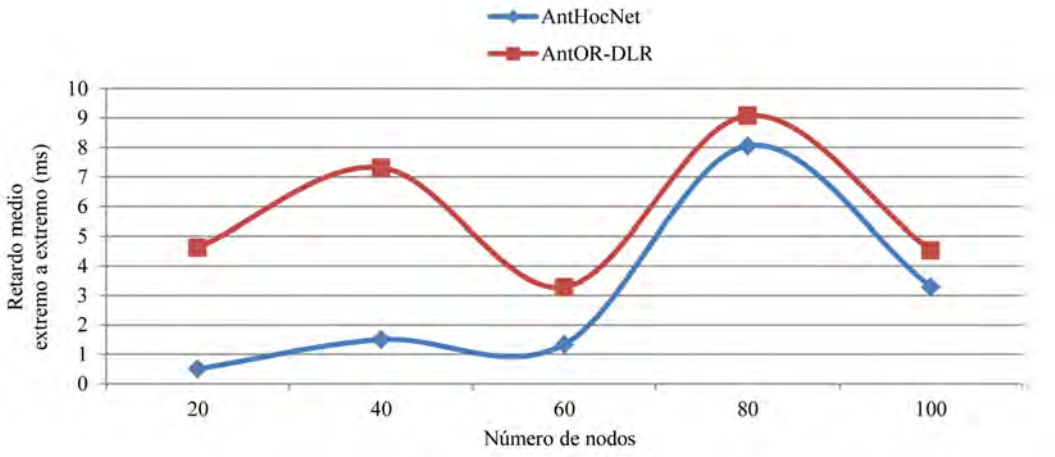


Figura 15.3: Retardo medio extremo a extremo (AntOR-DLR)

15.4.4 Sobrecarga en el Número de Paquetes

Como se observa en la Figura 15.4, la sobrecarga en el número de paquetes en AntOR es similar a la de AntHocNet en redes poco densas e inferior en redes densas, tanto más cuanto mayor sea el número de nodos existente. Esta disminución de la sobrecarga se explica porque al alcanzarse el número de nodos umbral para que el mecanismo disjunto de enlace sea efectivo, el protocolo tolera mucho mejor los fallos (al disponer de más rutas alternativas fiables) disminuyendo el número de paquetes de control RRFA en el proceso de reparación de rutas. Este hecho unido a lo observado en las métricas anteriores permite concluir que AntOR es especialmente escalable, al menos si lo comparamos con su predecesor AntHocNet.

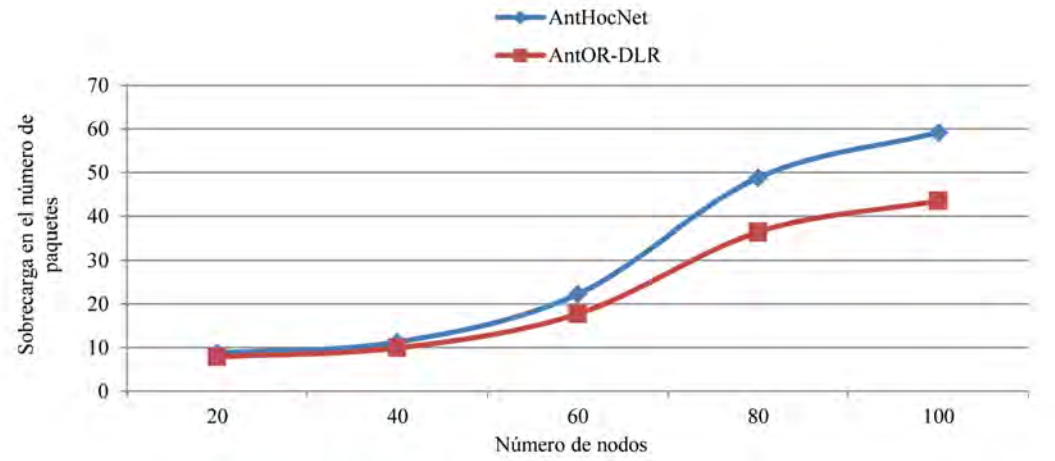


Figura 15.4: Sobrecarga en el número de paquetes (AntOR-DLR)

15.4.5 Sobrecarga en el Número de Bytes

Como se observa en la Figura 15.5, la sobrecarga en el número de bytes en AntOR tiene un comportamiento análogo a la sobrecarga en el número de paquetes, pudiendo concluirse lo señalado anteriormente.

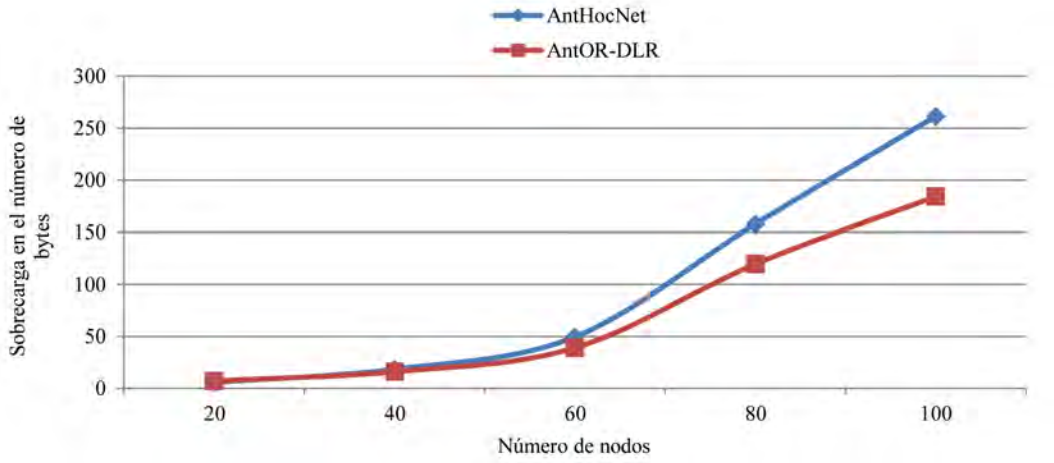


Figura 15.5: Sobrecarga en el número de bytes (AntOR-DLR)

15.5 Evaluación del Protocolo AntOR-DNR

Para evaluar las prestaciones del protocolo AntOR-DNR, en términos de efectividad, se ha tenido en cuenta el impacto del incremento del tiempo de pausa, esto es, cómo afecta éste a parámetros como el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo y el *jitter*. Esta evaluación se ha realizado conjuntamente con la del protocolo AntOR-DLR. Conviene reseñar que la variación del tiempo de pausa influye en el comportamiento del patrón de movilidad. Este incremento del tiempo de pausa tiene dos efectos en las propiedades generales de los escenarios relevantes para el encaminamiento. El primer efecto es que el decremento en la movilidad de los nodos (consecuencia de un tiempo de pausa alto) hace menos complicado el procesamiento del algoritmo de encaminamiento. El segundo efecto tiene que ver con la distribución de los nodos en el área del escenario cuando se utiliza el modelo de movilidad RWP. Se que ha comprobado que, según este modelo, hay una tendencia de los nodos a que aumente su densidad en el centro del área de la red y a que disminuya en los extremos, especialmente cuando es menor la movilidad.

15.5.1 Ratio de Entrega de Paquetes de Datos

Como se observa en la Figura 15.6, el ratio de entrega de paquetes de datos en AntOR-DLR es, en todo momento, superior al de AntOR-DNR, presentando además un comportamiento monótono más estable. Esto es debido a que AntOR-DLR es menos restrictivo (más tolerante) que AntOR-DNR (véase apartado 14.3.3.3). En otras palabras, es más fácil el cálculo de rutas disjuntas de enlace (toda ruta disjunta de nodo es de enlace pero no viceversa) y también más frecuente el fallo en rutas disjuntas de nodos (ya que la ruta de enlace disjunto, que se sirve de enlaces independientes, puede usar otros nodos).

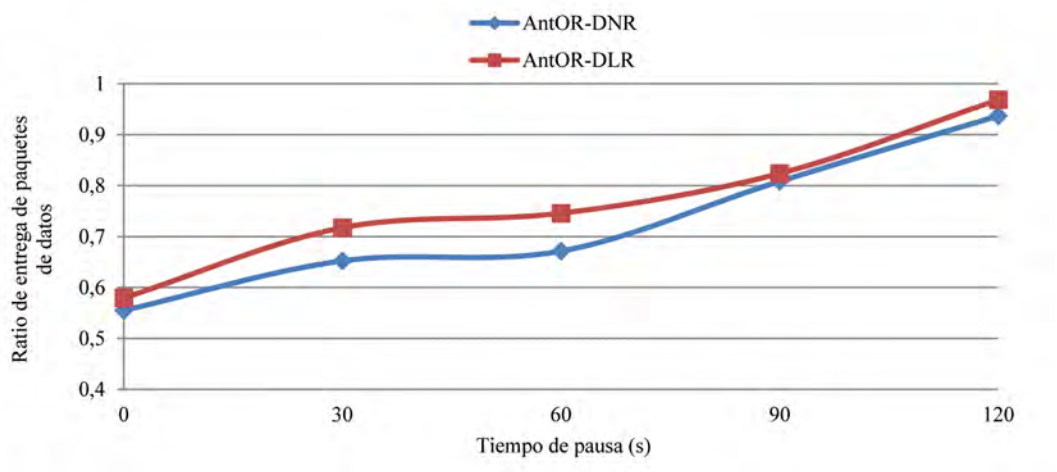


Figura 15.6: Ratio de entrega de paquetes de datos (AntOR-DNR)

15.5.2 Retardo Medio Extremo a Extremo

Como se observa en la Figura 15.7, el retardo medio extremo a extremo en AntOR-DLR es, en todo momento, inferior al de AntOR-DNR, presentando además un comportamiento monótono más estable. La explicación de este hecho es análoga a la realizada en el apartado anterior.

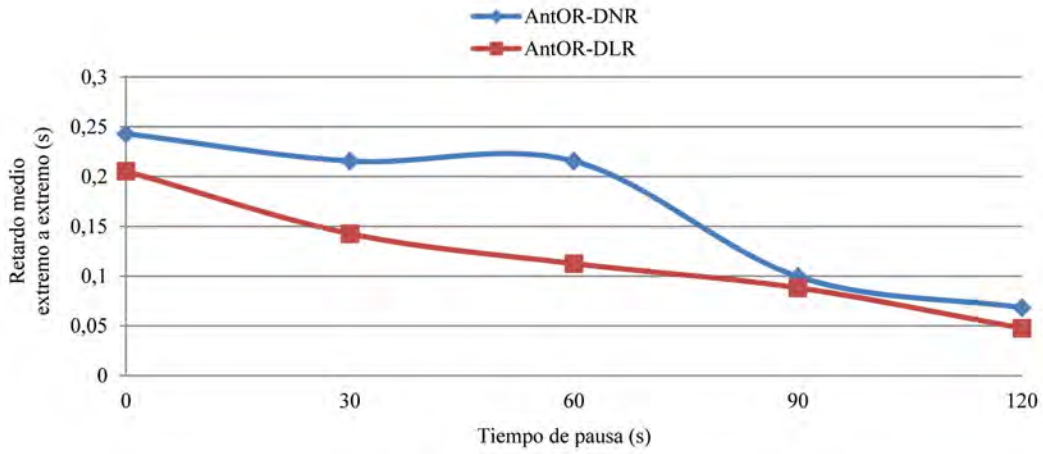


Figura 15.7: Retardo medio extremo a extremo (AntOR-DNR)

15.5.3 Jitter

Como se observa en la Figura 15.8, el *jitter* en AntOR-DLR es, en todo momento, inferior al de AntOR-DNR. A diferencia de las dos métricas anteriores la diferencia en el comportamiento monótono de ambos protocolos es más acentuada. Conviene recordar que el *jitter* es un parámetro que mide directamente la robustez (comportamiento frente a fallas) del algoritmo. Se concluye, por tanto, que la neutralización de fallos es mucho mejor en AntOR-DLR.

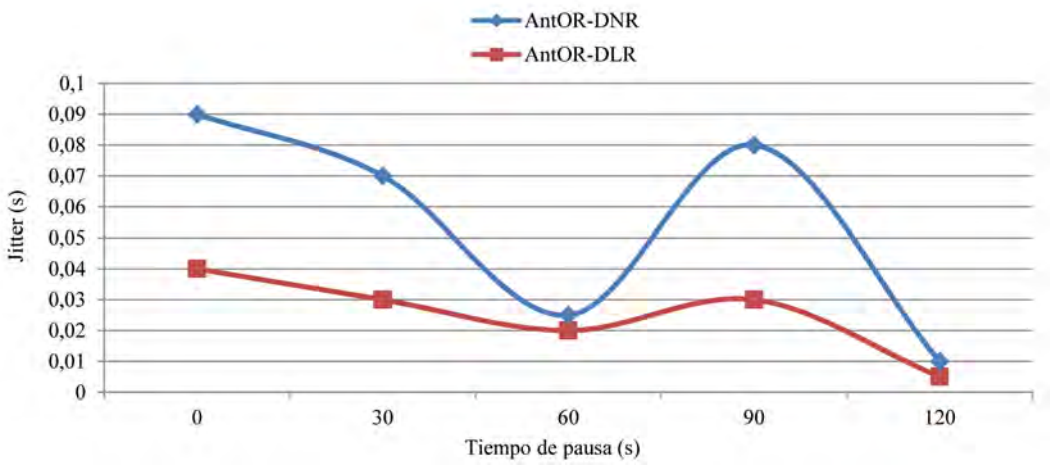


Figura 15.8: Jitter (AntOR-DNR)

15.6 Evaluación del Protocolo AntOR-RDLR

Para evaluar las prestaciones del protocolo AntOR-RDLR, en términos de efectividad, se ha tenido en cuenta el impacto del incremento del tiempo de pausa, esto es, cómo afecta éste a parámetros como el *throughput* y el ratio de entrega de paquetes de datos. Esta evaluación se ha realizado conjuntamente con la del protocolo AntOR-DLR. Previamente se analiza en este apartado cuáles son los valores idóneos del parámetro MAX_HOP del protocolo AntOR-RDLR.

15.6.1 Ajuste de MAX_HOP

Como se observa en la Figura 15.9, el valor óptimo de MAX_HOP, en términos de ratio de entrega de paquetes de datos, se alcanza para un valor de 6.

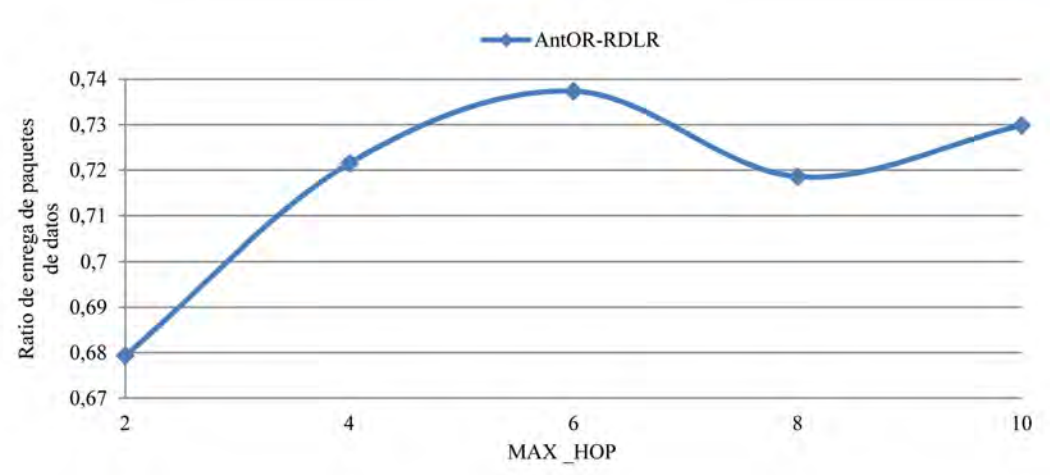


Figura 15.9: Ajuste de MAX_HOP - caso a (AntOR-RDLR)

Por otro lado, y como se observa en la Figura 15.10, el valor óptimo de MAX_HOP, en

términos de sobrecarga en el número de bytes, se alcanza para el mínimo valor (2 en este caso).

Consecuentemente, a la vista de las dos gráficas anteriores las mejores prestaciones de AntOR-RDLR se consiguen para valores de MAX_HOP en el intervalo [2, 6]. En la comparativa que sigue se ha elegido un valor de 5 para MAX_HOP.

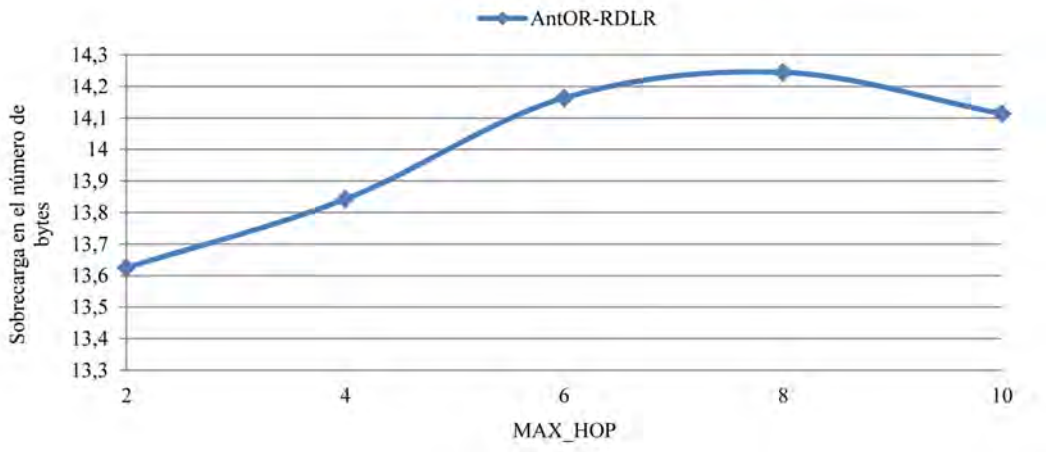


Figura 15.10: Ajuste de MAX_HOP - caso b (AntOR-RDLR)

15.6.2 Throughput

Como se observa en la Figura 15.11, el *throughput* en AntOR-RDLR es, en todo momento, superior al de su predecesor, AntOR-DLR. Esta mejora del *throughput* es consecuencia de la mayor tolerancia a fallos de AntOR-RDLR, debido a la posibilidad de disponer de más rutas alternativas usando nodos que pertenecen a la ruta principal, hecho que no ocurre en AntOR-DLR.

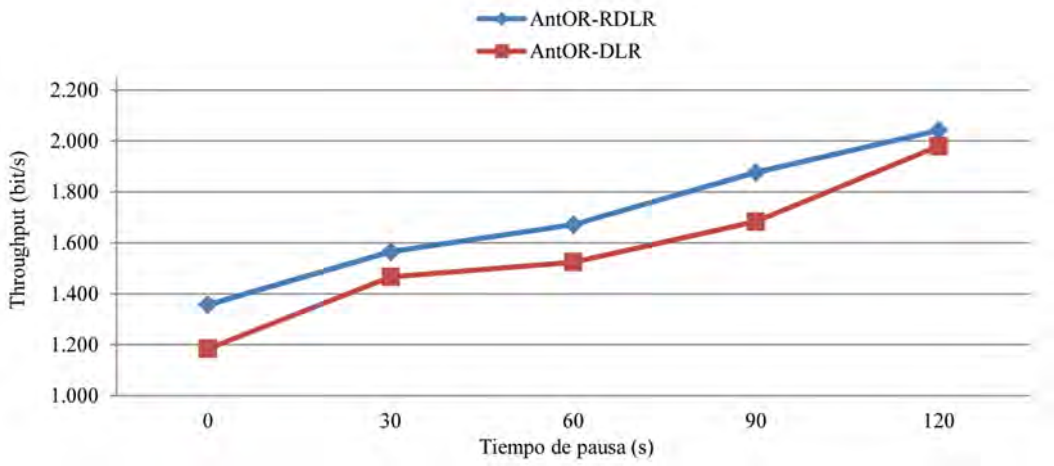


Figura 15.11: Throughput (AntOR-RDLR)

15.6.3 Ratio de Entrega de Paquetes de Datos

Como se observa en la Figura 15.12, el ratio de entrega de paquetes en AntOR-RDLR es, en todo momento, superior al de su predecesor. La explicación de este hecho es análoga a la realizada en el apartado anterior.

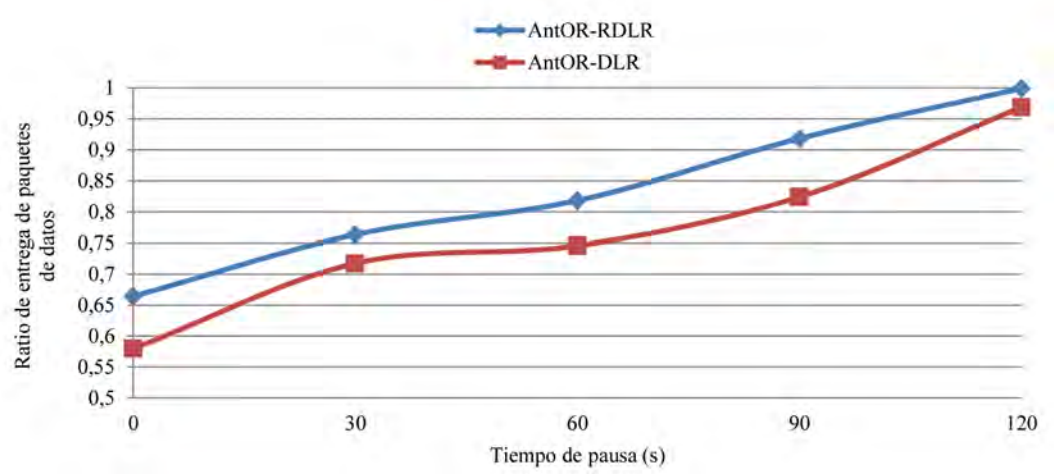


Figura 15.12: Ratio de entrega de paquetes de datos (AntOR-RDLR)

15.7 Evaluación del Protocolo AntOR-UDLR

Para evaluar las prestaciones del protocolo AntOR-UDLR, tanto en términos de eficiencia como de efectividad, se ha tenido en cuenta el impacto del incremento del tiempo de pausa y cómo afecta éste a parámetros como el *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, la sobrecarga en el número de paquetes. Asimismo, se ha tenido en cuenta el impacto del incremento de la velocidad de los nodos y cómo afecta éste a parámetros como *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, la sobrecarga en el número de bytes. Esta evaluación se ha realizado conjuntamente con la de los protocolos AntOR-DLR y OLSR.

15.7.1 Throughput

Como se observa en la Figura 15.13, el *throughput* en AntOR-UDLR es, en todo momento, superior al de su predecesor, AntOR-DLR, independientemente del tiempo de pausa.

Análogamente, como se observa en la Figura 15.14, el *throughput* en AntOR-UDLR es también, en todo momento, superior al de su predecesor, AntOR-DLR, independientemente de la velocidad de los nodos.

De lo anterior puede concluirse que las modificaciones introducidas en AntOR-ULDR mejoran la efectividad del protocolo. En otras palabras, el proceso de neutralización de fallos de enlace se realiza más rápidamente en AntOR-UDLR gracias al envío de paquetes unicast, más fiables que los paquetes broadcast utilizados por su predecesor.

Asimismo, conviene señalar la gran diferencia entre AntOR-UDLR / AntOR-DLR respecto a OLSR, diferencia que se amplía ostensiblemente en escenarios muy dinámicos, lo que se explica fácilmente por el carácter proactivo de este último.

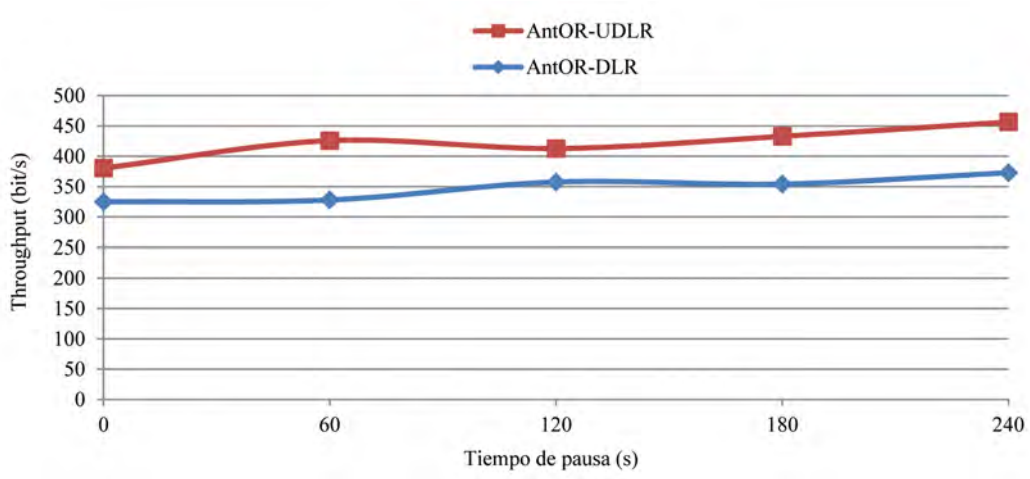


Figura 15.13: Throughput - caso a (AntOR-UDLR)

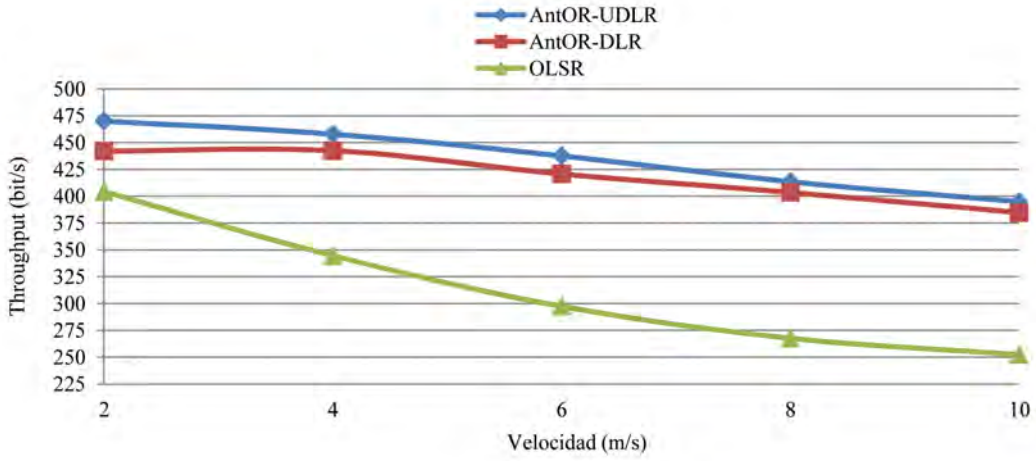


Figura 15.14: Throughput - caso b (AntOR-UDLR)

15.7.2 Ratio de Entrega de Paquetes de Datos

Como se observa en las Figuras 15.15 y 15.16, el ratio de entrega de paquetes de datos en AntOR-UDLR es, en todo momento, superior al de AntOR-DLR. Asimismo, tanto AntOR-UDLR como AntOR-DLR mejoran a OLSR. La explicación de este hecho es análoga a la realizada en el apartado anterior.

15.7.3 Retardo medio extremo a extremo

Como se observa en las Figuras 15.17 y 15.18, el retardo medio extremo a extremo en AntOR-UDLR es, en todo momento, menor que AntOR-DLR, siendo además más uniforme. Este último hecho permite concluir las buenas propiedades de escalabilidad de AntOR-UDLR. Asimismo, se observa cómo el retardo en OLSR es aún menor que en ambos protocolos. Esto es debido a que OLSR, al ser puramente proactivo, presenta una baja latencia. Conviene recordar que los protocolos diseñados en esta Tesis son híbridos, estando estas diferencias en los valores normales que se encuentran en la literatura.

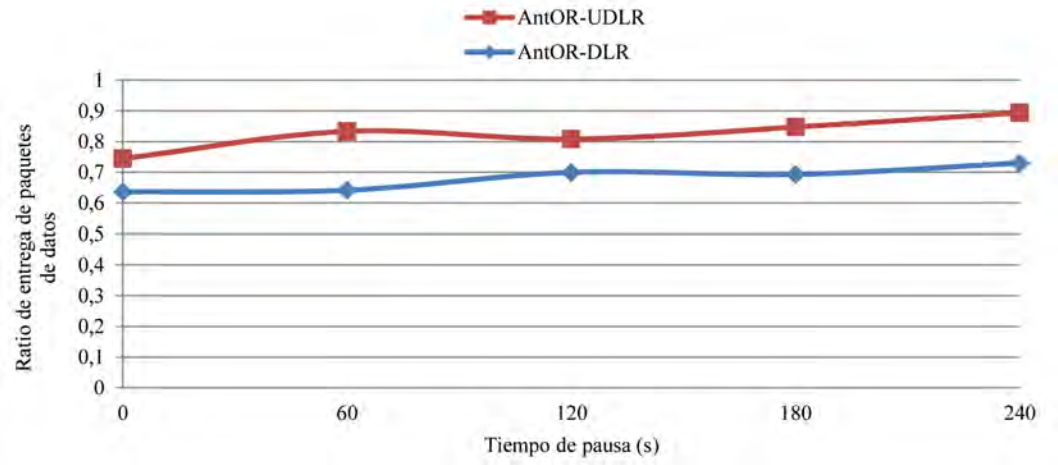


Figura 15.15: Ratio de entrega de paquetes de datos - caso a (AntOR-UDLR)

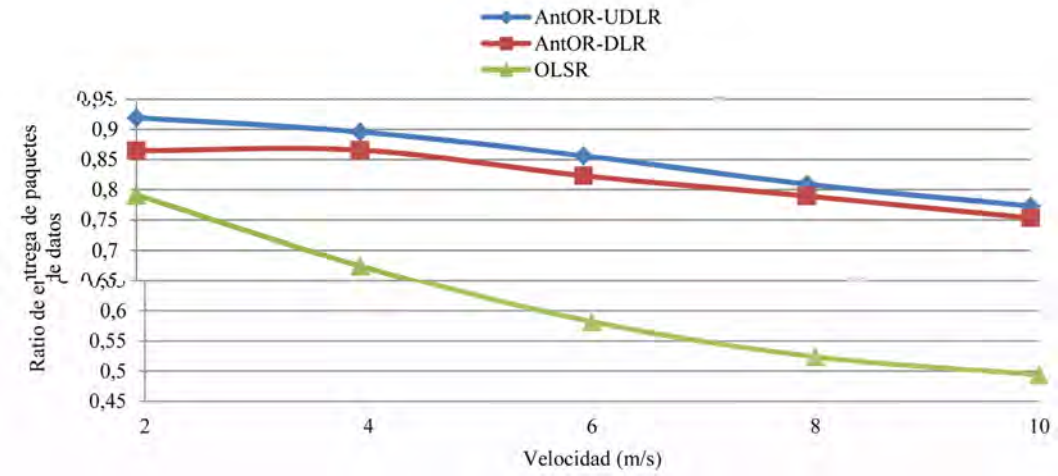


Figura 15.16: Ratio de entrega de paquetes de datos - caso b (AntOR-UDLR)

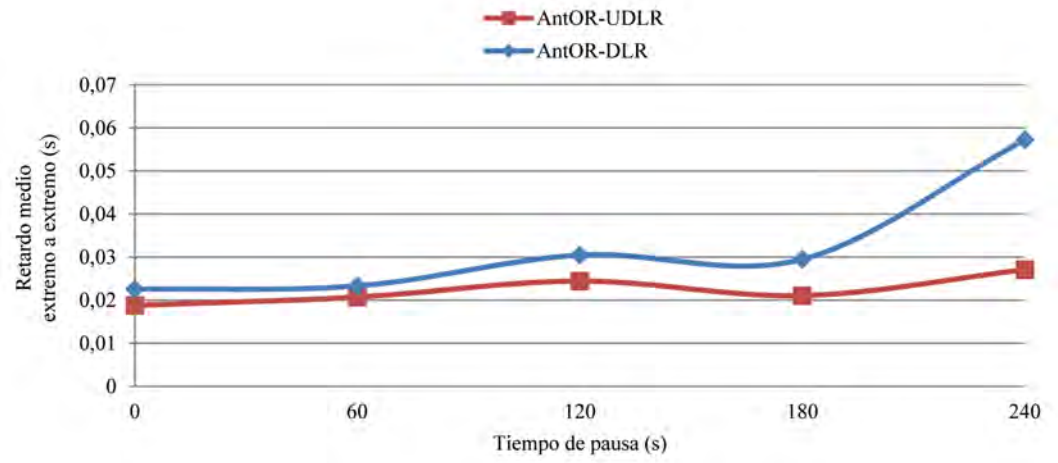


Figura 15.17: Retardo medio extremo a extremo - caso a (AntOR-UDLR)

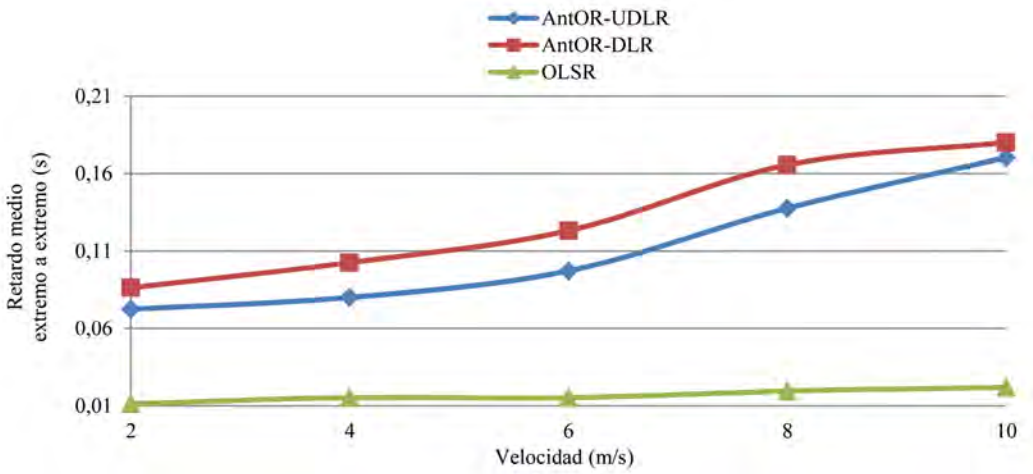


Figura 15.18: Retardo medio extremo a extremo - caso b (AntOR-UDLR)

15.7.4 Sobrecarga en el Número de Paquetes

Como se observa en la Figura 15.19, la sobrecarga en el número de paquetes en AntOR-UDLR es, en términos generales, similar a la de AntOR-DLR.

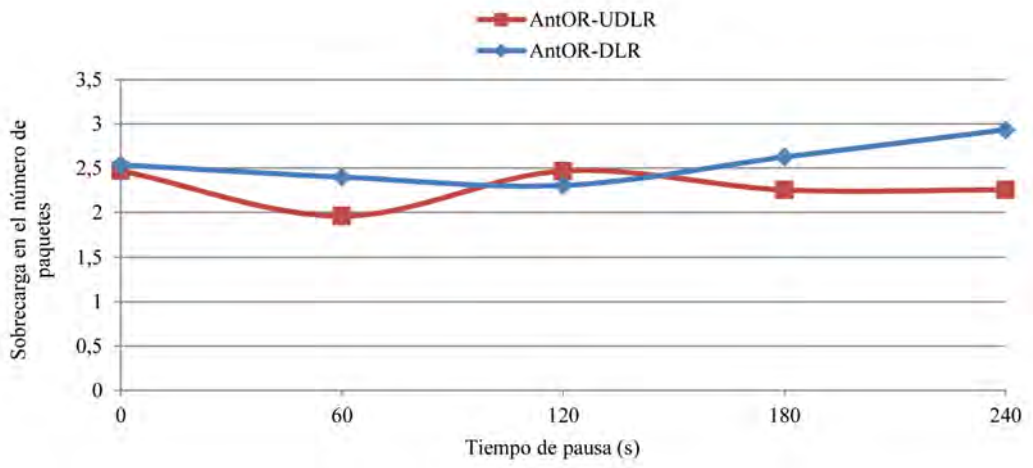


Figura 15.19: Sobrecarga en el número de paquetes (AntOR-UDLR)

15.7.5 Sobrecarga en el Número de Bytes

Como se observa en la Figura 15.20, e idénticamente a lo comentado en el apartado anterior, la sobrecarga en el número de bytes en AntOR-UDLR es, en términos generales, similar a la de AntOR-DLR. Asimismo, se observa que estas sobrecargas son significativamente menores que las de OLSR, lo cual es lógico dado el carácter proactivo de éste.

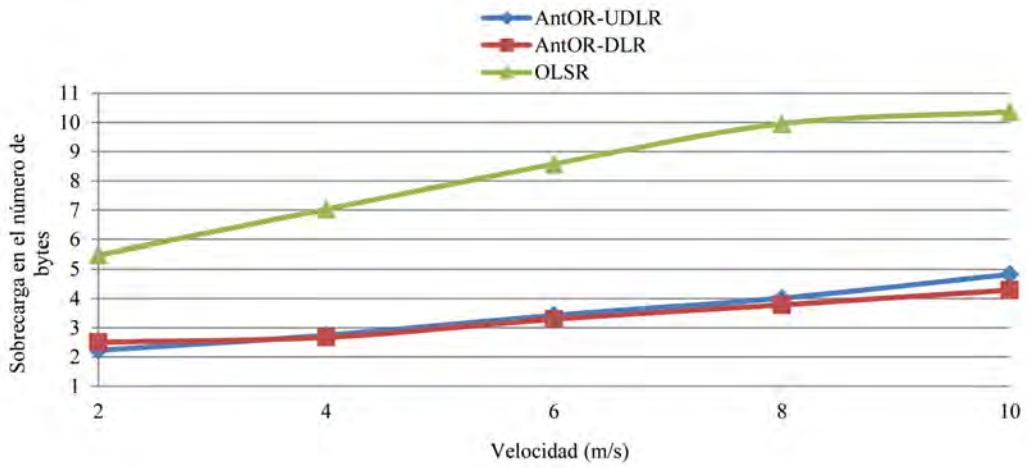


Figura 15.20: Sobrecarga en el número de bytes (AntOR-UDLR)

15.8 Evaluación del Protocolo AntOR-v2

Para evaluar las prestaciones del protocolo AntOR-v2, tanto en términos de eficiencia como de efectividad, se ha tenido en cuenta el impacto del incremento del tiempo de pausa y cómo afecta éste a parámetros como el ratio de entrega de paquetes de datos, el *jitter* y la sobrecarga en el número de paquetes. Asimismo, se ha tenido en cuenta el impacto del incremento del número de nodos y cómo afecta éste a parámetros como el *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, el *jitter*, la sobrecarga en el número de paquetes y la sobrecarga en el número de bytes. Esta evaluación se ha realizado conjuntamente con la del protocolo AODV. Se ha elegido AODV por dos razones: en primer lugar, la mayoría de los protocolos híbridos se comparan en la literatura con él (es una referencia obligada); en segundo lugar, se ha elegido para este protocolo una comparativa con un protocolo reactivo como AODV ya que anteriormente la comparativa se ha hecho con un protocolo proactivo como OLSR.

15.8.1 Throughput

Como se observa en la Figura 15.21, el *throughput* en AntOR-v2 es, en todo momento, superior al de AODV, independientemente del número de nodos, siendo especialmente significativa la diferencia entre ambos en redes densas. Asimismo, el *throughput* en AntOR-v2 decae lentamente en este tipo de redes. Ambos hechos determinan un buen comportamiento de AntOR-v2 respecto a la escalabilidad de la red.

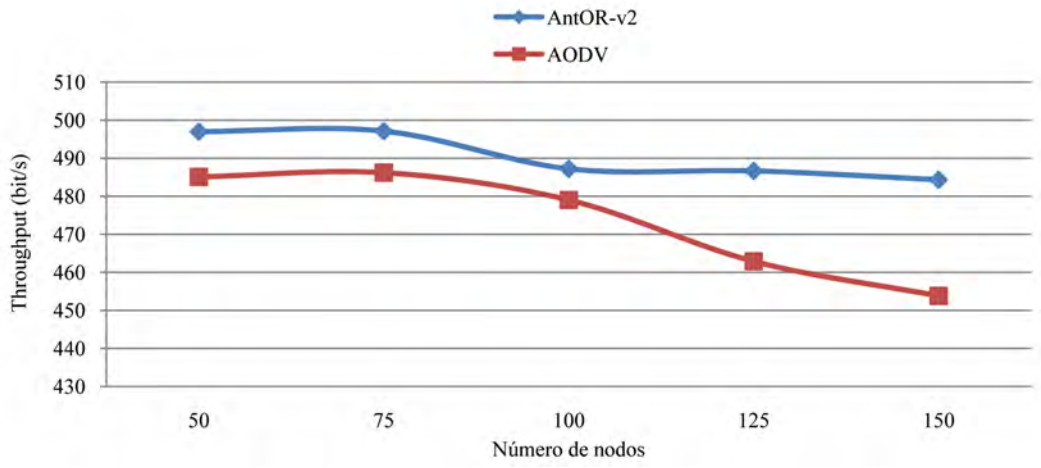


Figura 15.21: Throughput (AntOR-v2)

15.8.2 Ratio de Entrega de Paquetes de Datos

Como se observa en las Figuras 15.22 y 15.23, y análogamente a lo comentado en el apartado anterior, el ratio de entrega de paquetes de datos en AntOR-v2 es, en todo momento, superior al de AODV, independientemente del número de nodos, siendo especialmente significativa la diferencia entre ambos en redes densas. Asimismo, el ratio de entrega de paquetes de datos en AntOR-v2 decae lentamente en este tipo de redes. Ambos hechos determinan un buen comportamiento de AntOR-v2 respecto a la escalabilidad de la red.

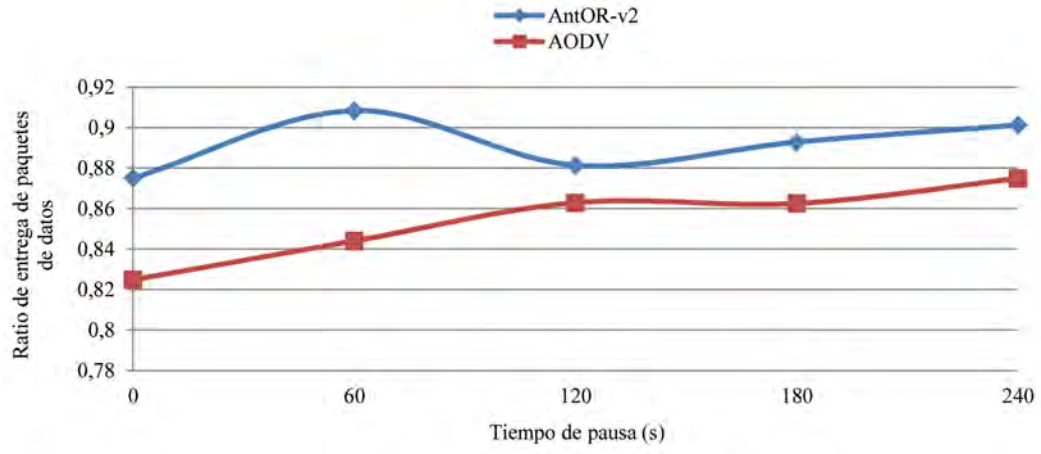


Figura 15.22: Ratio de entrega de paquetes de datos - caso a (AntOR-v2)

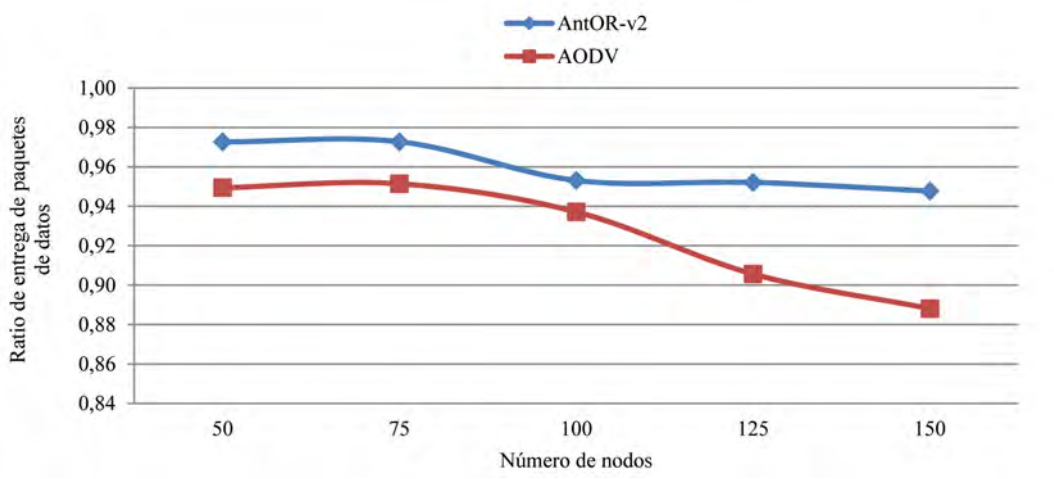


Figura 15.23: Ratio de entrega de paquetes de datos - caso b (AntOR-v2)

15.8.3 Retardo Medio Extremo a Extremo

Como se observa en la Figura 15.24, el retardo medio extremo a extremo en AntOR-v2 es, en todo momento, inferior al de AODV. Esto es lógico dado el carácter reactivo de AODV.

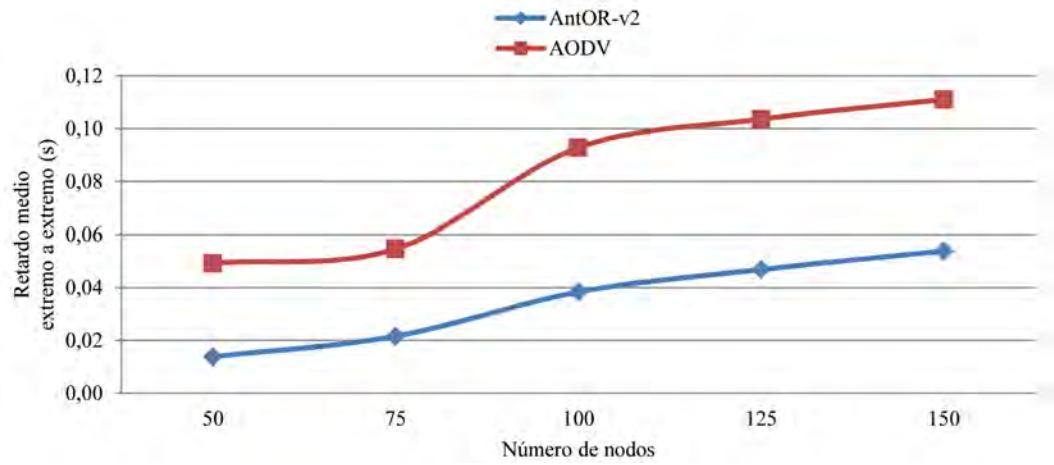


Figura 15.24: Retardo medio extremo a extremo (AntOR-v2)

15.8.4 Jitter

Como se observa en la Figura 15.25, el *jitter* en AntOR-v2 es, en términos generales, inferior al de AODV y prácticamente constante, independientemente del tiempo de pausa, lo que hace de AntOR-v2 un protocolo bastante robusto. Asimismo, y como se observa en la Figura 15.26, el *jitter* en AntOR-v2 es, en todo momento, claramente inferior al de AODV, independientemente del número de nodos, presentando un comportamiento monótono similar.

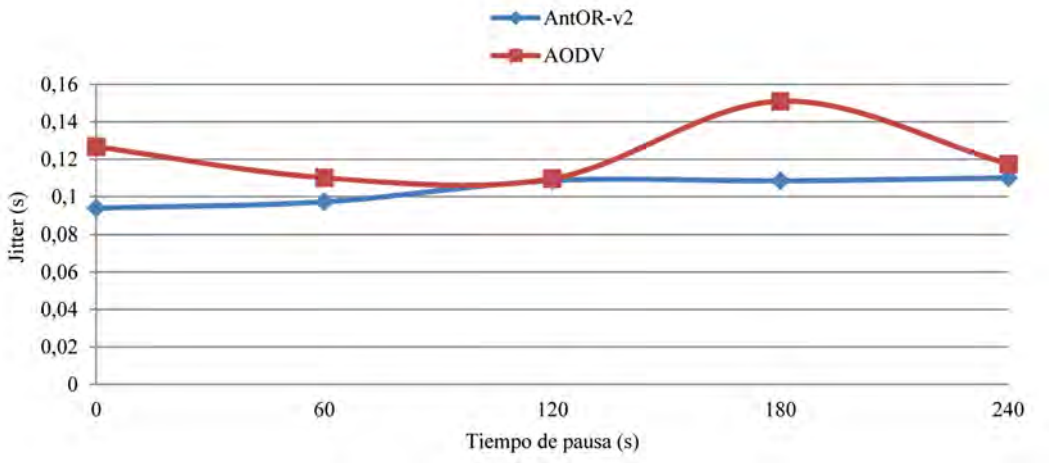


Figura 15.25: Jitter - caso a (AntOR-v2)

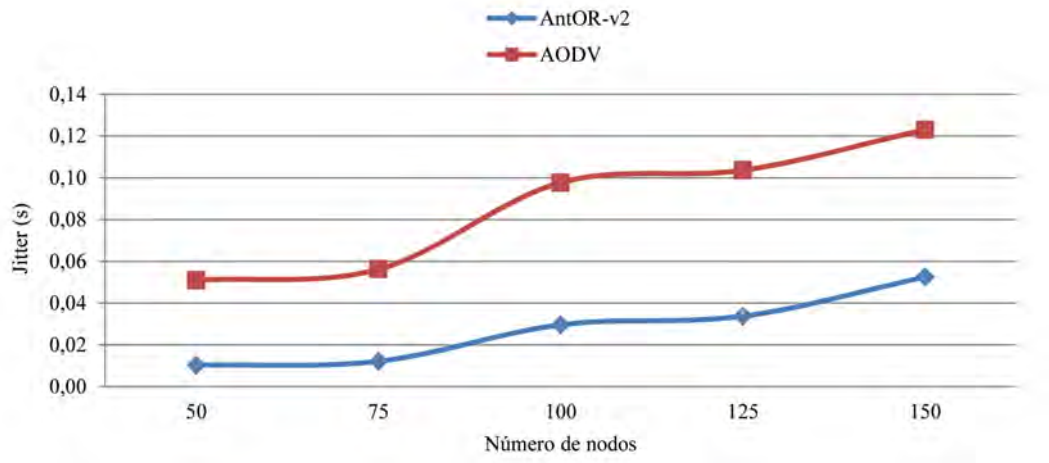


Figura 15.26: Jttter - caso b (AntOR-v2)

15.8.5 Sobrecarga en el Número de Paquetes

Como se observa en la Figura 15.27, la Sobrecarga en el Número de Paquetes en AntOR-v2 es muy similar a la de AODV, independientemente del tiempo de pausa. En cambio, y como se observa en la Figura 15.28, la sobrecarga en AntOR-v2 es, en todo momento, ligeramente superior a la de AODV, independientemente del número de nodos, estrechándose la diferencia en redes densas.

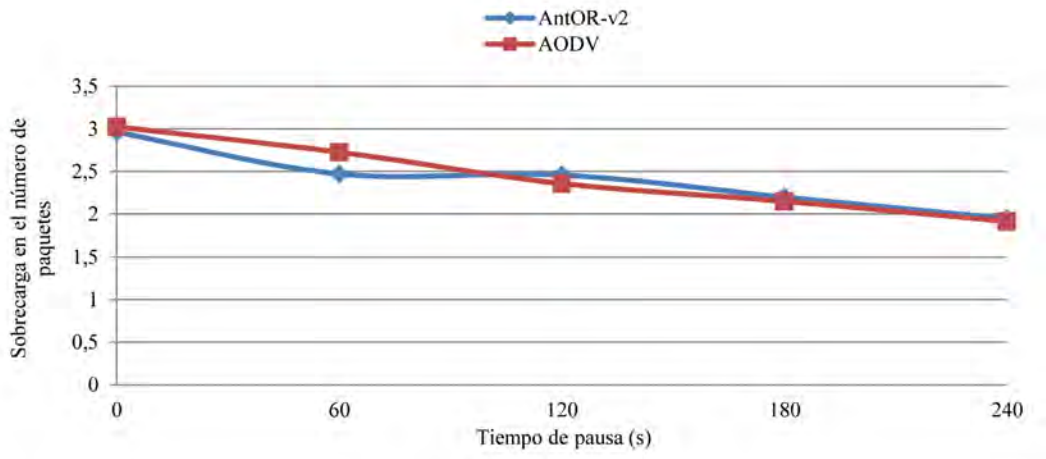


Figura 15.27: Sobrecarga en el número de paquetes - caso a (AntOR-v2)

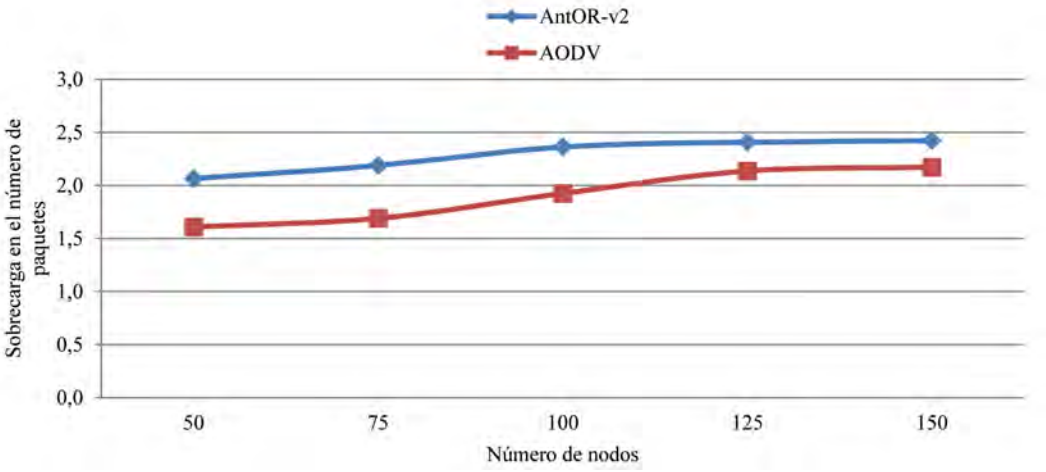


Figura 15.28: Sobrecarga en el número de paquetes - caso b (AntOR-v2)

15.8.6 Sobrecarga en el Número de Bytes

Como se observa en la Figura 15.29, y análogamente a lo comentado en la Figura 15.28, la sobrecarga en AntOR-v2 es, en todo momento, ligeramente superior a la de AODV, independientemente del número de nodos, estrechándose la diferencia en redes densas.

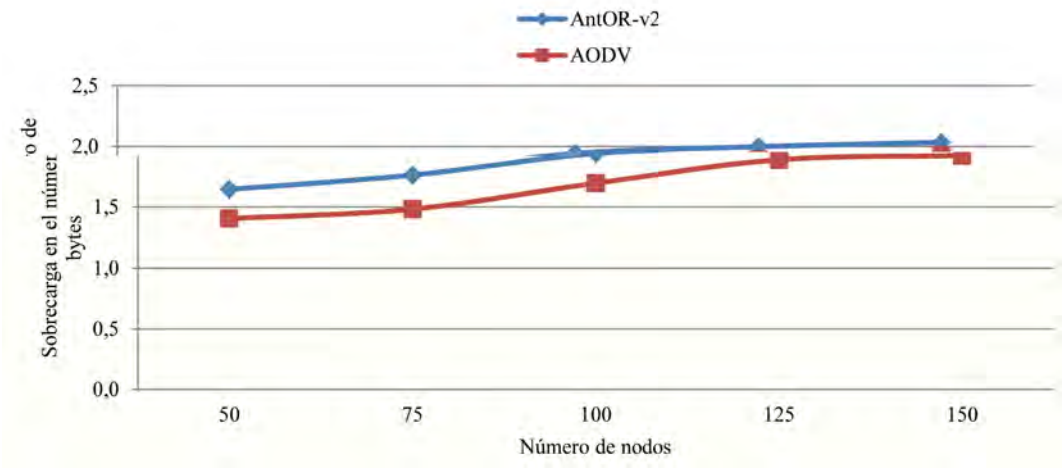


Figura 15.29: Sobrecarga en el número de bytes (AntOR-v2)

15.9 Evaluación del Protocolo HACOR

Para evaluar las prestaciones del protocolo HACOR, tanto en términos de eficiencia como de efectividad, se ha tenido en cuenta el impacto del incremento del tiempo de pausa y cómo afecta éste a parámetros como el *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, el *jitter*, la sobrecarga en el número de paquetes y la sobrecarga en el número de bytes. Asimismo, se ha tenido en cuenta el impacto del incremento del número de nodos y cómo afecta éste a parámetros como el *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, el *jitter*, la sobrecarga en el número de paquetes y la sobrecarga en el número de bytes. Esta evaluación se ha realizado conjuntamente con la de los estándares AODV y OLSR.

15.9.1 Throughput

Como se observa en las Figuras 15.30 y 15.31, el *throughput* en HACOR es, en todo momento, superior al de los otros dos protocolos. Asimismo, apenas decae con el número de nodos, lo que permite concluir su buena predisposición para la escalabilidad.

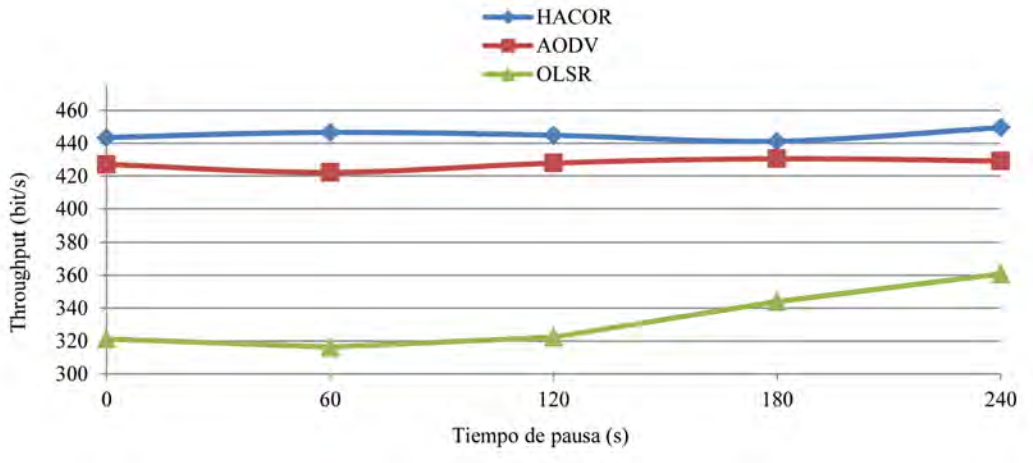


Figura 15.30: Throughput - caso a (HACOR)

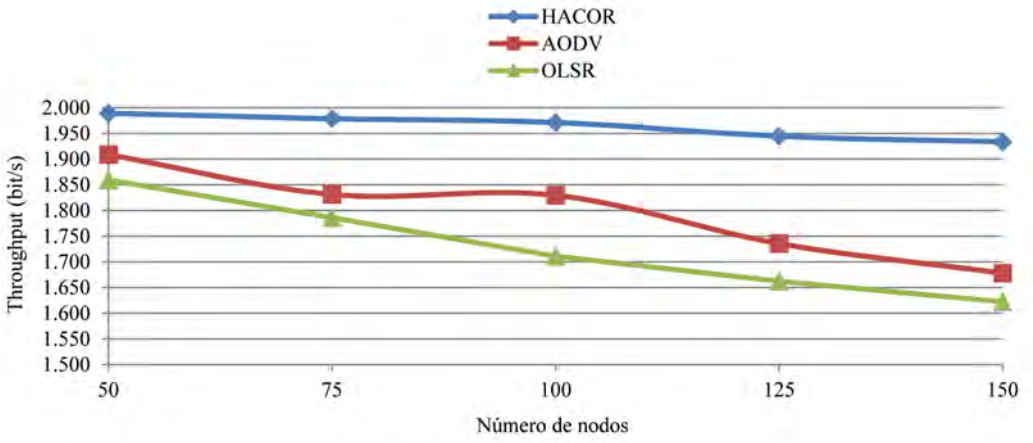


Figura 15.31: Throughput - caso b (HACOR)

15.9.2 Ratio de Entrega de Paquetes de Datos

Como se observa en las Figuras 15.32 y 15.33, y análogamente a lo comentado en el apartado anterior, el ratio de entrega de paquetes de datos en HACOR es, en todo momento, superior al de los otros dos protocolos. Asimismo, apenas decae con el número de nodos, lo que permite reafirmar su buena predisposición para la escalabilidad.

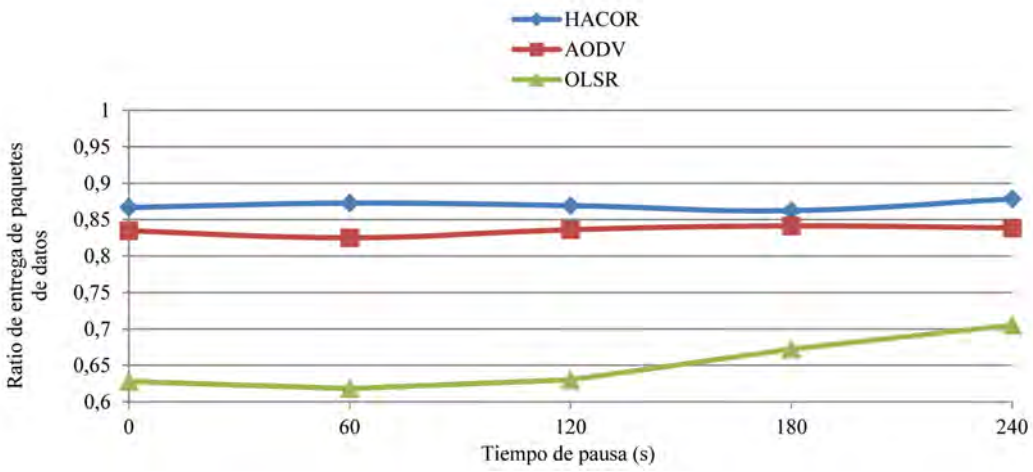


Figura 15.32: Ratio de entrega de paquetes de datos - caso a (HACOR)

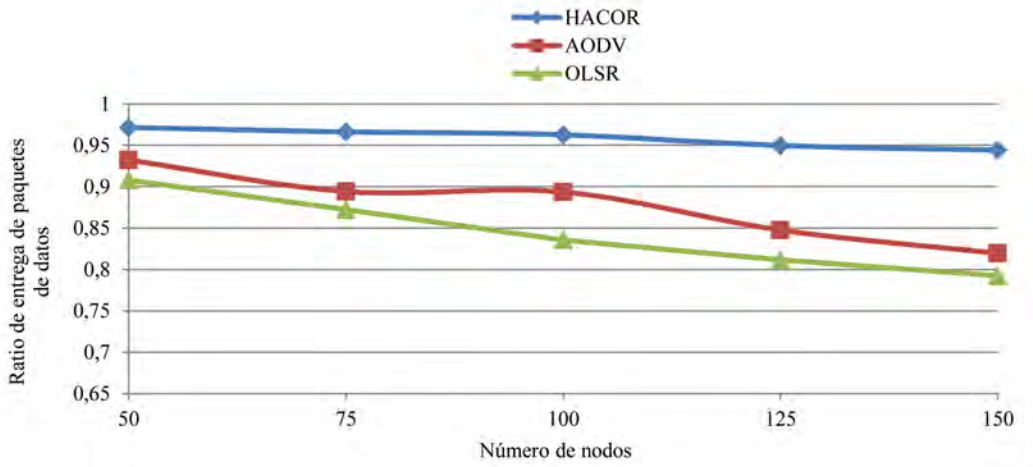


Figura 15.33: Ratio de entrega de paquetes de datos - caso b (HACOR)

15.9.3 Retardo Medio Extremo a Extremo

Como se observa en las Figuras 15.34 y 15.35, el retardo medio extremo a extremo en HACOR toma valores intermedios a los de AODV y OLSR. Esto es lógico puesto que la latencia de un protocolo híbrido habitualmente oscila entre la de un reactivo y un proactivo. No obstante, y como se observa en la Figura 15.35, el retardo es en HACOR ligeramente superior al de OLSR, independientemente del número de nodos. Esto último es especialmente reseñable puesto que HACOR minimiza considerablemente el retardo medio extremo a extremo, mostrando su carácter proactivo y ocultando, en cierto modo, su carácter reactivo. Al igual que en las otras dos métricas analizadas anteriormente se puede concluir su excelente predisposición para la escalabilidad.

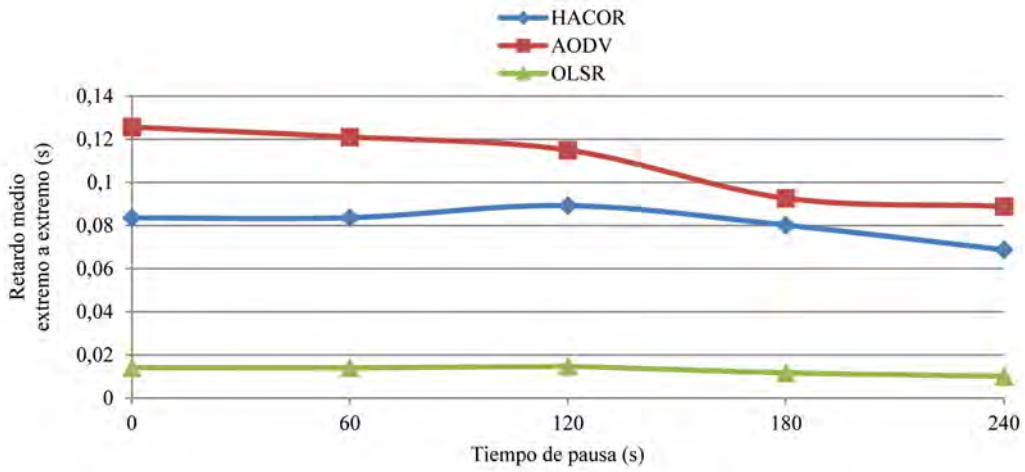


Figura 15.34: Retardo medio extremo a extremo - caso a (HACOR)

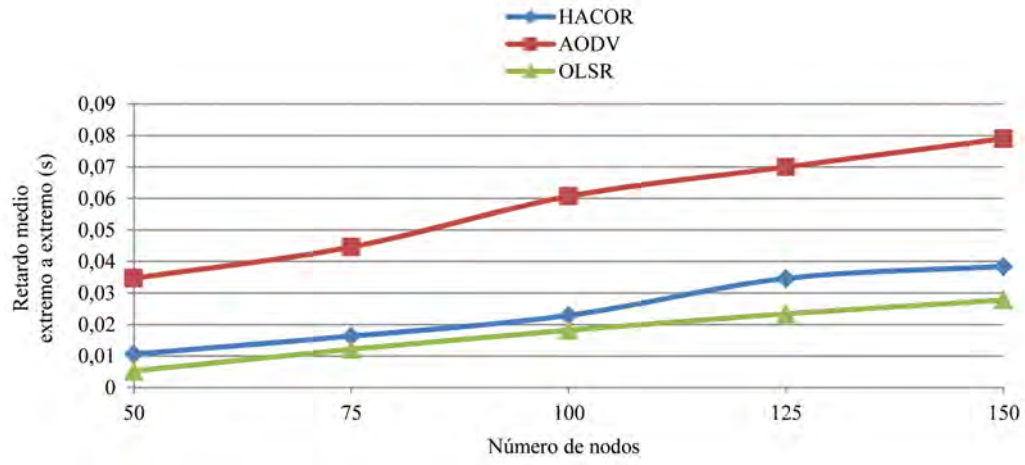


Figura 15.35: Retardo medio extremo a extremo - caso b (HACOR)

15.9.4 Jitter

Como se observa en la Figura 15.36, y en cierta forma de forma análoga a lo comentado en el apartado anterior, el *jitter* en HACOR toma valores intermedios a los de AODV y OLSR con respecto al tiempo de pausa. Asimismo, y como se observa en la Figura 15.37, HACOR es el más robusto presentando valores similares de *jitter*, independientemente del número de nodos, mejorando a partir de un umbral en el número de nodos a AODV, circunstancia que se acentúa en redes particularmente densas. Al igual que en las métricas analizadas anteriormente se puede concluir su excelente predisposición para la escalabilidad.

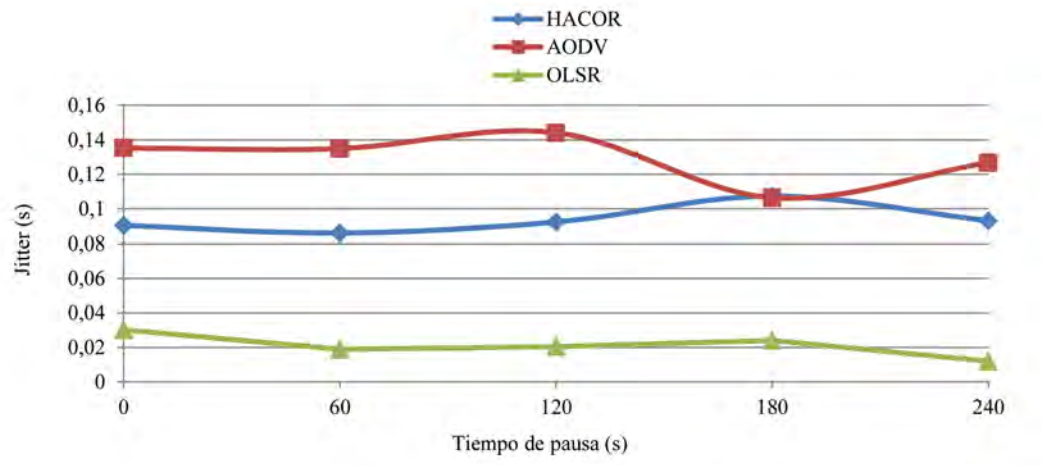


Figura 15.36: Jitter - caso a (HACOR)

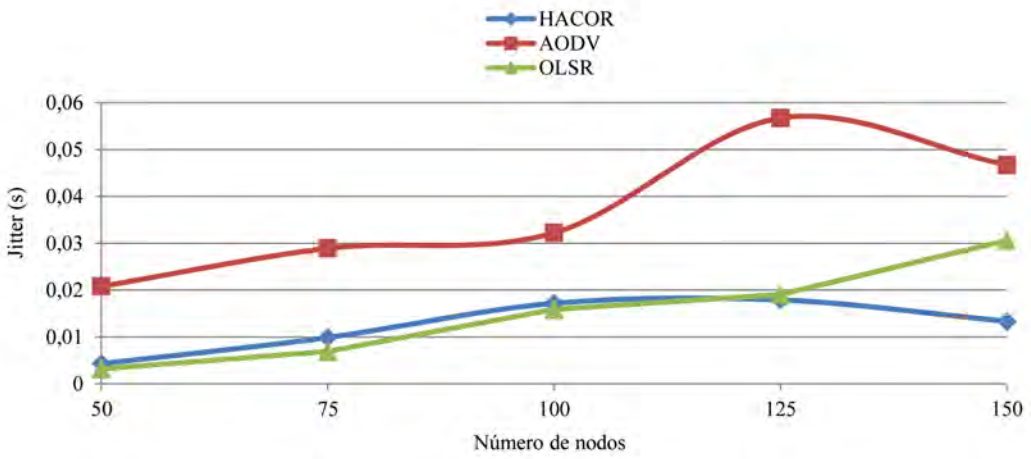


Figura 15.37: Jitter - caso b (HACOR)

15.9.5 Sobrecarga en el Número de Paquetes

Como se observa en la Figura 15.38, la sobrecarga en el número de paquetes en HACOR toma valores intermedios a los de AODV y OLSR con respecto al tiempo de pausa. Sin embargo, y como se observa en la Figura 15.39, la sobrecarga es superior a la de AODV y OLSR con respecto al número de nodos. Conviene reseñar como aspecto favorable que en redes densas las diferencias con OLSR se estrechan hasta llegar a ser nulas o prácticamente nulas. Puede concluirse que el precio a pagar por las métricas analizadas anteriormente es un ligero incremento en la sobrecarga, precio que disminuye conforme aumenta el número de nodos, lo que también en cierta forma afirma su buena predisposición para la escalabilidad.

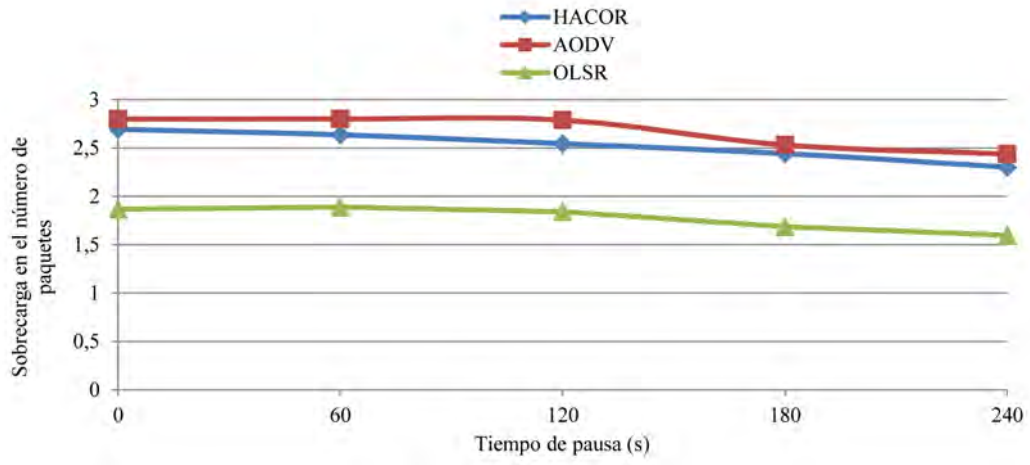


Figura 15.38: Sobrecarga en el número de paquetes - caso a (HACOR)

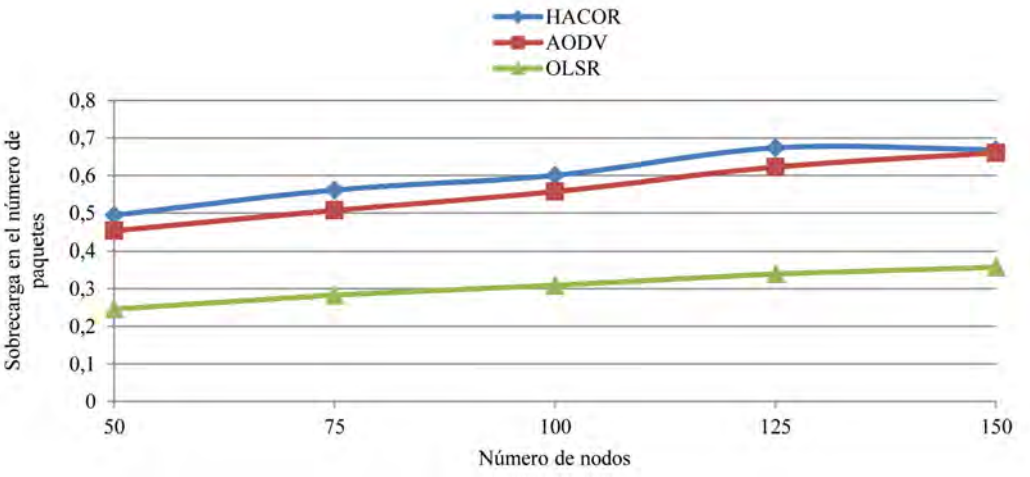


Figura 15.39: Sobrecarga en el número de paquetes - caso b (HACOR)

15.9.6 Sobrecarga en el Número de Bytes

Como se observa en las Figuras 15.40 y 15.41, la sobrecarga en el número de bytes en AntOR tiene un comportamiento análogo a la sobrecarga en el número de paquetes, pudiendo concluirse lo señalado anteriormente.

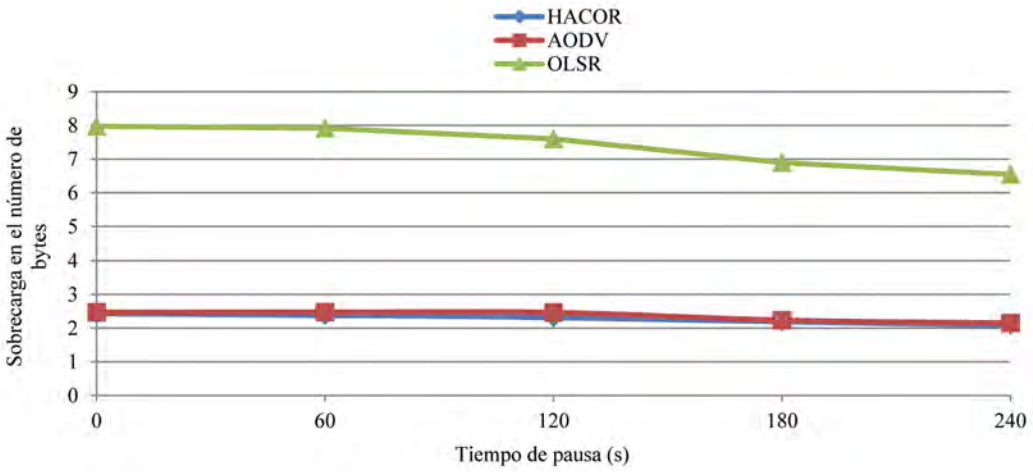


Figura 15.40: Sobrecarga en el número de bytes - caso a (HACOR)

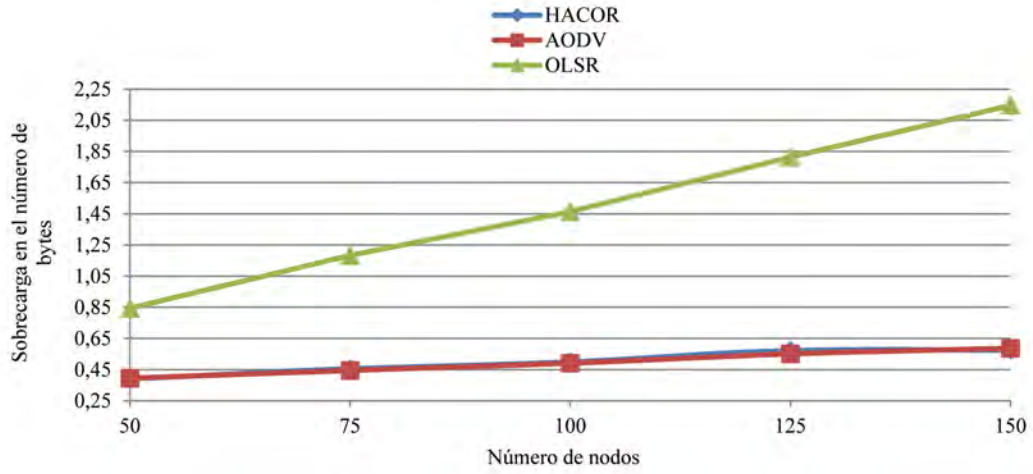


Figura 15.41: Sobrecarga en el número de bytes - caso b (HACOR)

15.10 Evaluación del Protocolo PAntOR

Para evaluar las prestaciones del protocolo PAntOR, tanto en términos de eficiencia como de efectividad, se ha tenido en cuenta el impacto del incremento de la velocidad de los nodos y cómo afecta éste a parámetros como el *throughput*, el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo, el *jitter* y la sobrecarga en el número de paquetes. Asimismo, se ha tenido en cuenta el impacto del incremento del tiempo de pausa y cómo afecta éste a parámetros como el ratio de entrega de paquetes de datos, el retardo medio extremo a extremo y el *jitter*. Esta evaluación se ha realizado conjuntamente con la de su protocolo predecesor, AntOR-DNR.

15.10.1 Throughput

Como se observa en la Figura 15.42, el *throughput* en PAntOR es superior, en todo momento, al de AntOR-DNR. Esto se explica fácilmente por la paralelización introducida en la fase de establecimiento de ruta y en los procesos de reparación local de ruta y notificación de fallos de enlace.

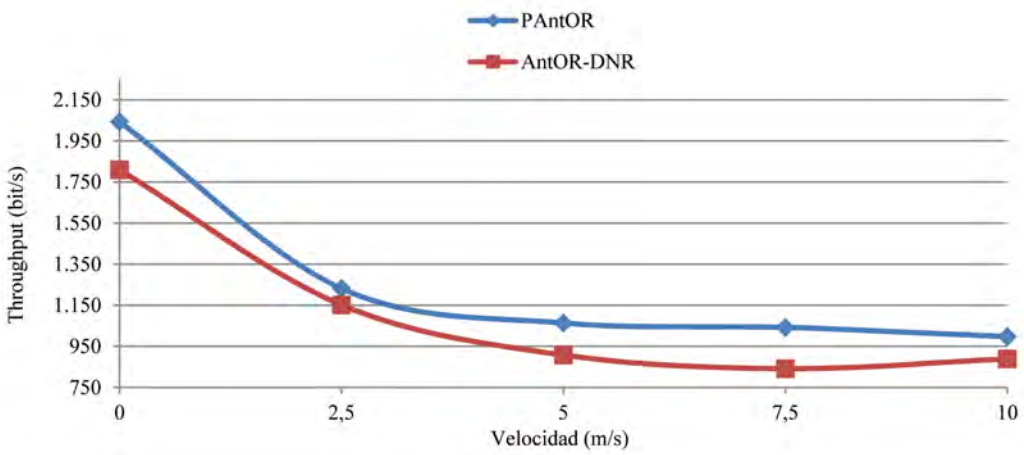


Figura 15.42: Throughput (PAntOR)

15.10.2 Ratio de Entrega de Paquetes de Datos

Como se observa en las Figuras 15.43 y 15.44, el ratio de entrega de paquetes de datos en PAntOR es superior, en todo momento, al de AntOR-DNR. Como se ha señalado anteriormente, esto se explica fácilmente por la paralelización introducida en la fase de establecimiento de ruta y en los procesos de reparación local de ruta y notificación de fallos de enlace.

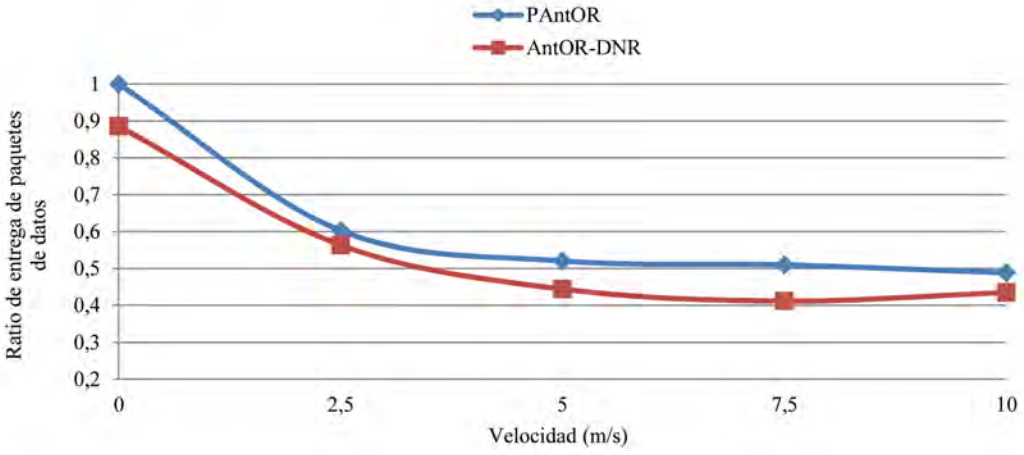


Figura 15.43: Ratio de entrega de paquetes de datos - caso a (PAntOR)

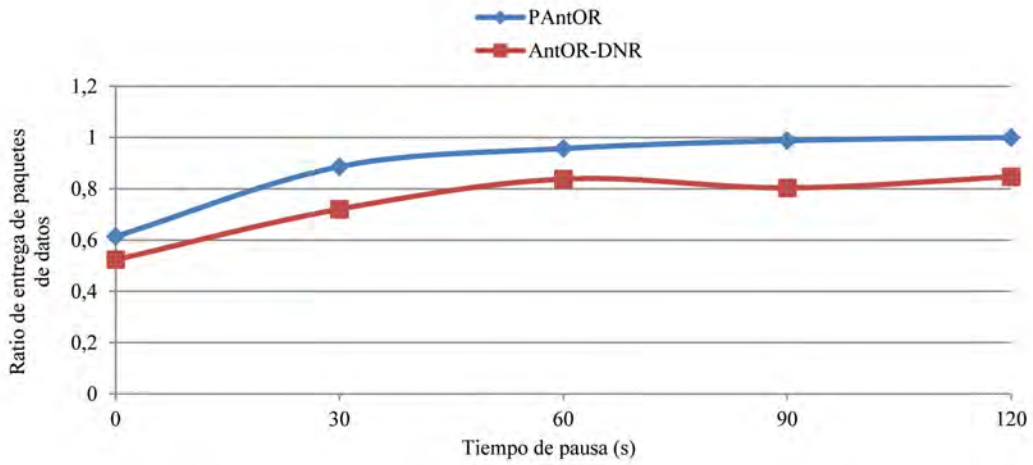


Figura 15.44: Ratio de entrega de paquetes de datos - caso b (PAntOR)

15.10.3 Retardo Medio Extremo a Extremo

Como se observa en las Figuras 15.45 y 15.46, el retardo medio extremo a extremo en PAntOR es, en todo momento, inferior al de su predecesor. Esta mejora es consecuencia de la paralelización realizada, especialmente de la introducida en la fase de establecimiento de ruta (fase 1 del protocolo).

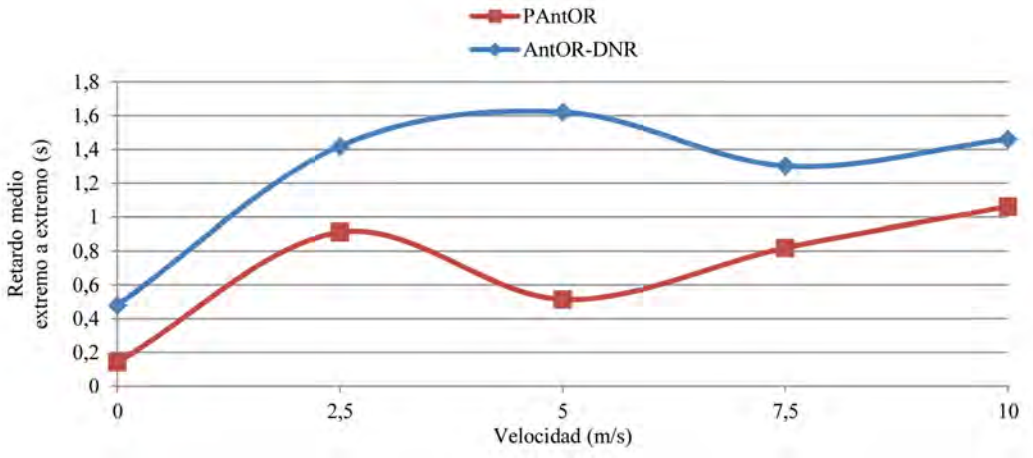


Figura 15.45: Retardo medio extremo a extremo - caso a (PAntOR)

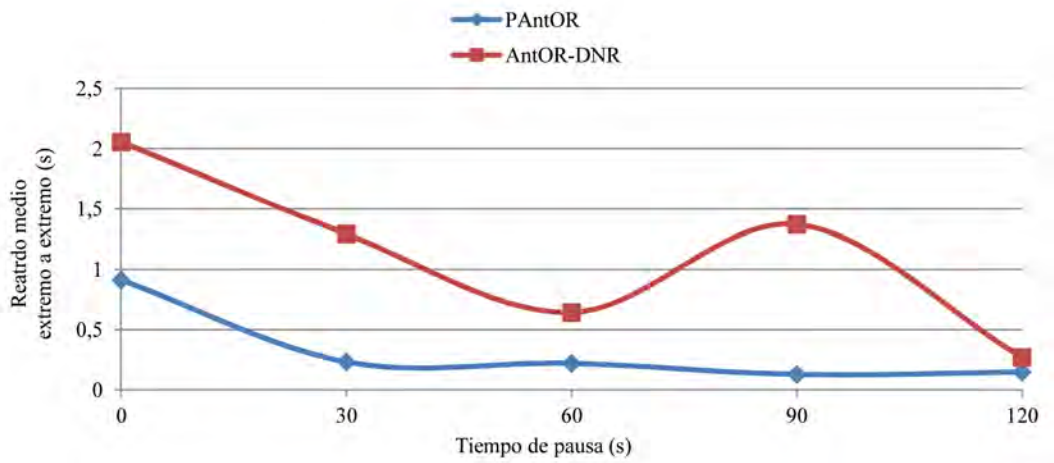


Figura 15.46: Retardo medio extremo a extremo - caso b (PAntOR)

15.10.4 Jitter

Como se observa en las Figuras 15.47 y 15.48, el *jitter* en PAntOR es inferior, en todo momento, al de AntOR-DNR. Esta mejora es consecuencia de la paralelización realizada, especialmente de la introducida en los procesos de reparación local de ruta y notificación de fallos de enlace (fase 4 del protocolo).

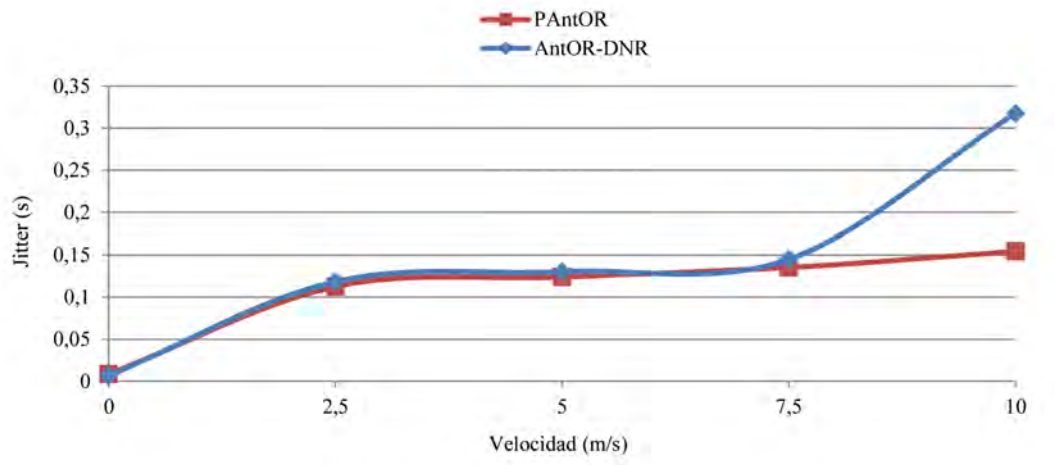


Figura 15.47: Jitter - caso a (PAntOR)

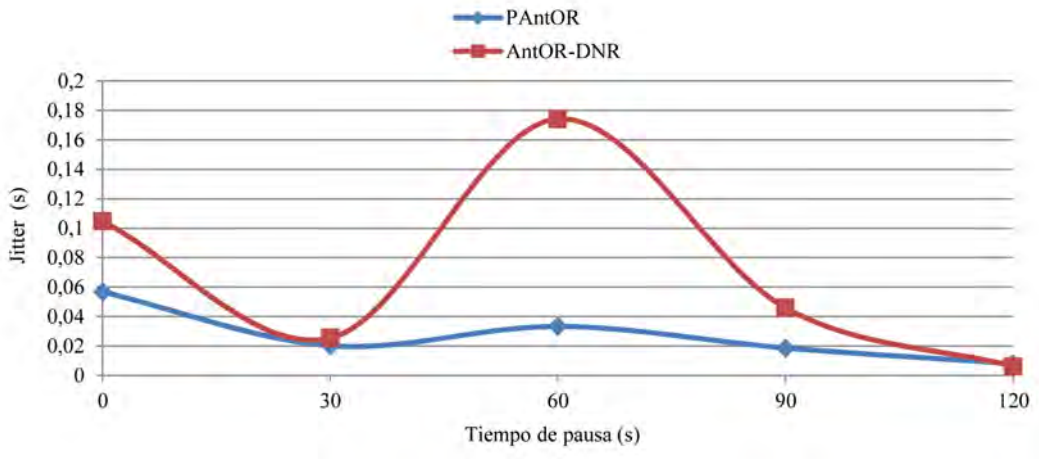


Figura 15.48: Jitter - caso b (PAntOR)

15.10.5 Sobrecarga en el Número de Paquetes

Como se observa en la Figura 15.49, la sobrecarga en el número de paquetes en PAntOR es, en todo momento, inferior a la de su predecesor. Asimismo, se observa cómo esta diferencia se incrementa con la velocidad de los nodos (o con la rotura de enlaces). Esto se explica fácilmente porque PAntOR agiliza los procesos de reparación local de ruta, evitando además el modo broadcast ya que los agentes son enviados a los vecinos del nodo local que percibe el fallo.

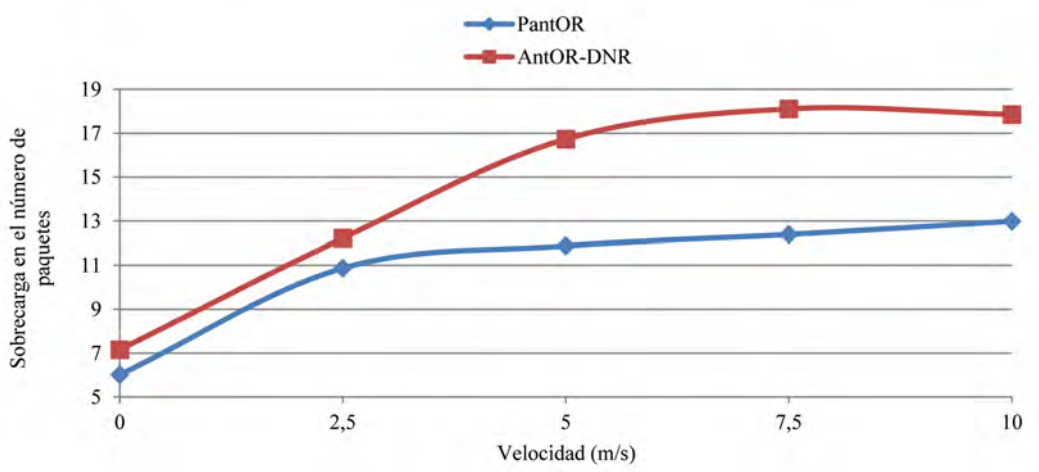


Figura 15.49: Sobrecarga en el número de paquetes (PAntOR)

15.11 Evaluación del Protocolo PAntOR-MI

Para evaluar las prestaciones del protocolo PAntOR-MI, en términos de efectividad, se ha tenido en cuenta el impacto del incremento de la velocidad de los nodos y cómo afecta éste al ratio de entrega de paquetes de datos. Esta evaluación se ha realizado conjuntamente con la de los protocolos PAntOR y AntOR-DNR.

15.11.1 Ratio de Entrega de Paquetes de Datos

Como se observa en la Figura 15.50, el ratio de entrega de paquetes de datos en PAntOR-MI es mejor que el de su predecesor en entornos dinámicos. Esto se explica fácilmente porque PAntOR-MI paraleliza la fase de establecimiento de ruta (fase 1 del protocolo) mediante hilos con el uso multi-interfaz, perdiéndose menos paquetes de datos.

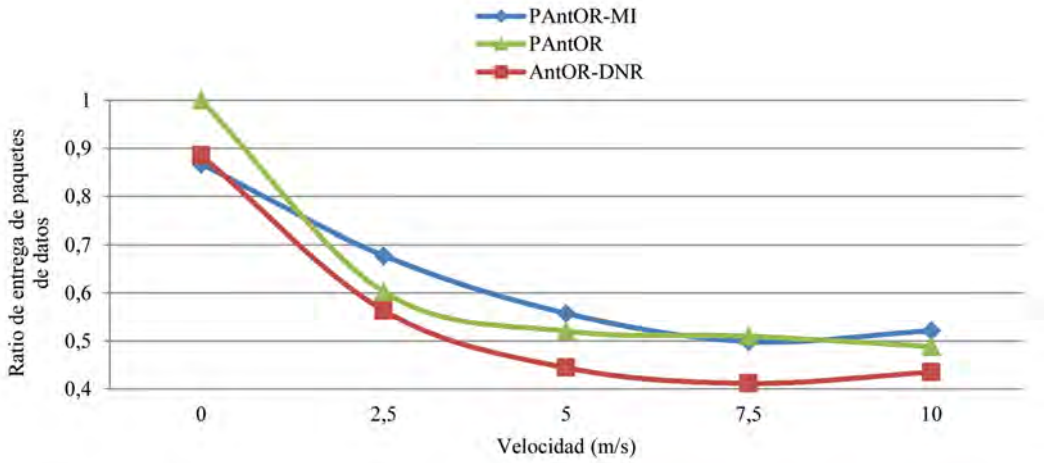


Figura 15.50: Ratio de entrega de paquetes de datos (PAntOR-MI)

15.12 Resumen

Este capítulo ha recogido las simulaciones realizadas considerando escenarios reales con el objeto de verificar la aplicabilidad de AntOR, AntOR-DLR, AntOR-DNR, AntOR-RDLR, AntOR-UDLR, AntOR-v2, HACOR, PAntOR y PAntOR-MI. Para ello se ha utilizado el simulador de redes NS-3 [NS3]

En AntOR-DLR se han evaluado las siguientes métricas: throughput, ratio de entrega de paquetes de datos, retardo medio extremo a extremo, sobrecarga en el número de paquetes y sobrecarga en el número de bytes. Los resultados de las simulaciones demuestran que el protocolo AntOR-DLR presenta respecto a AntHocNet mejor *throughput* y mejor ratio de entrega de paquetes de datos así como un pequeño incremento del retardo medio extremo a extremo y de la sobrecarga, aspectos negativos éstos últimos que en redes densas se igualan.

En AntOR-DNR se han evaluado las siguientes métricas: ratio de entrega de paquetes de datos, retardo medio extremo a extremo y *jitter*. Los resultados de las simulaciones demuestran que el protocolo AntOR-DLR mejora a AntOR-DNR en todas las métricas analizadas.

En AntOR-RDLR se han evaluado las siguientes métricas: *throughput* y ratio de entrega de paquetes de datos. Los resultados de las simulaciones demuestran que el protocolo AntOR-RDLR mejora a AntOR-DLR en todas las métricas analizadas.

En AntOR-UDLR se han evaluado las siguientes métricas: *throughput*, ratio de entrega de paquetes de datos, retardo medio extremo a extremo, sobrecarga en el número de paquetes y sobrecarga en el número de bytes. Los resultados de las simulaciones demuestran que el protocolo AntOR-UDLR mejora a su predecesor en todas las métricas analizadas, a excepción de la sobrecarga, en cuyo caso ambos protocolos presentan valores similares.

En AntOR-v2 se han evaluado las siguientes métricas: *throughput*, ratio de entrega de paquetes de datos, retardo medio extremo a extremo, *jitter*, sobrecarga en el número de paquetes y sobrecarga en el número de bytes. Los resultados de las simulaciones demuestran que el protocolo AntOR-v2 mejora a AODV en todas las métricas analizadas, a excepción de la sobrecarga que es ligeramente superior, diferencia ésta que se hace imperceptible en redes densas.

En HACOR se han evaluado las siguientes métricas: *throughput*, ratio de entrega de paquetes de datos, retardo medio extremo a extremo, *jitter*, sobrecarga en el número de paquetes y sobrecarga en el número de bytes. Los resultados de las simulaciones demuestran que el protocolo HACOR mejora a AODV y OLSR en el throughput y en el ratio de entrega de paquetes de datos, presentando valores intermedios respecto a estos dos protocolos en las métricas de retardo extremo a extremo, *jitter*, sobrecarga en el número de paquetes y sobrecarga en el número de bytes.

En PAntOR se han evaluado las siguientes métricas: *throughput*, ratio de entrega de paquetes de datos, retardo medio extremo a extremo, *jitter* y sobrecarga en el número de paquetes. Los resultados de las simulaciones demuestran que el protocolo PAntOR mejora a AntOR-DNR en todas las métricas analizadas.

En PAntOR-MI se ha evaluado la siguiente métrica: ratio de entrega de paquetes de datos. Los resultados de las simulaciones demuestran que el protocolo PAntOR-MI mejora a su predecesor en entornos muy dinámicos.

Capítulo 16

Conclusiones y Trabajo Futuro

Este trabajo ha abordado un aspecto fundamental de las denominadas redes móviles ad hoc como es la problemática del encaminamiento.

En primer lugar se ha visto la necesidad del diseño de protocolos de encaminamiento específicos para las redes móviles ad hoc dada la naturaleza de las mismas, así como las características o requisitos que deben cumplir para funcionar adecuadamente, comentándose también la imposibilidad de utilizar las soluciones tradicionales.

En segundo lugar se ha analizado un grupo de algoritmos o protocolos de encaminamiento denominados bioinspirados que tienen como característica esencial el hecho de ser adaptativos, algo especialmente reseñable en este tipo de ambientes. Dentro de estos algoritmos han sido especialmente referenciados en la literatura los basados en el concepto de inteligencia colectiva, esto es, aquellos que aplican el comportamiento social de los insectos y de otros animales para resolver problemas. El algoritmo Ant Colony Optimization (ACO) o algoritmo de optimización de la colonia de hormigas constituye el punto de partida de estos algoritmos. Los algoritmos ACO se basan en el comportamiento colectivo de las hormigas en su búsqueda del alimento. ACO se aplica a una amplia gama de problemas diferentes. Debido a sus propiedades de adaptabilidad y robustez, también se ha convertido en un paradigma para el encaminamiento en redes móviles ad hoc. Los algoritmos ACO trabajan de forma iterativa. En cada paso las hormigas artificiales construyen en paralelo una solución para el problema en cuestión, utilizando la matriz de feromona artificial. A continuación, se actualiza la matriz de feromona sobre la base de las soluciones encontradas. De esta manera, la matriz de feromona refleja información sobre las buenas soluciones que se han encontrado hasta la fecha, y permite a las hormigas de las generaciones posteriores utilizar esta información para crear otras nuevas.

En tercer lugar se ha realizado una revisión del estado del arte de los protocolos de encaminamiento ACO para redes móviles ad hoc observándose que no existen protocolos representativos cuyas métricas de funcionamiento se degraden poco o nada en entornos escalables. Esta revisión ha incluido un análisis exhaustivo del protocolo AntHocNet, la referencia indiscutible en el área. El estudio de la literatura ha recogido también una recopilación de las principales técnicas de paralelización de los algoritmos ACO, algo especialmente interesante si se pretende dar una solución escalable.

Posteriormente, se ha especificado un nuevo protocolo de encaminamiento ACO para redes móviles ad hoc denominado AntOR. Como su predecesor AntHocNet, AntOR es híbrido en el sentido de que contiene elementos de encaminamiento tanto reactivos como proactivos. En concreto, combina un proceso reactivo de establecimiento de ruta con un proceso proactivo de mantenimiento y exploración de nuevas rutas. La información de encaminamiento se almacena en tablas de feromona que son similares a las utilizadas por

otros algoritmos de encaminamiento ACO. El reenvío de paquetes de datos y de control se realiza de una manera estocástica con el uso de estas tablas. Los fallos de enlace se tratan con mecanismos reactivos específicos, tales como la reparación local de ruta y el uso de mensajes de aviso. Los aspectos clave del protocolo AntOR son la utilización de rutas disjuntas de nodo y disjuntas de enlace, la separación entre la feromona regular y la feromona virtual en el proceso de difusión y la exploración de nuevas rutas, que tiene en cuenta el número de saltos en las mejores rutas.

A continuación se ha especificado una familia de protocolos de encaminamiento ACO para redes móviles ad hoc. Todos ellos derivan del protocolo AntOR, que presenta dos variantes: la versión disjunta de enlace (AntOR-DLR) y la versión disjunta de nodo (AntOR-DNR). La versión disjunta de enlace ha dado lugar a un conjunto de protocolos secuenciales: AntOR-RDLR, AntOR-UDLR, AntOR-v2 y HACOR. Todos estos protocolos son refinamientos sucesivos del protocolo original. La versión disjunta de nodo ha dado lugar a un conjunto de protocolos paralelos: PAntOR y PAntOR-MI.

En AntOR-DNR las rutas no comparten nodos y en AntOR-DLR no comparten enlaces. Por la propiedad disjunta un fallo en un nodo afecta a una ruta y no a toda la red. Además, el balanceo de carga es mejor (al no repetirse las rutas). El cálculo de rutas en AntOR-DLR es más fácil (menos restrictivo) que en AntOR-DNR ya que toda ruta disjunta de nodo es también disjunta de enlace, pero no al contrario.

AntOR-RDLR se diferencia de su predecesor (AntOR-DLR) en el proceso de actualización de feromonas y en el mecanismo de descubrimiento de rutas, que permite a las hormigas proactivas hacia delante ir por rutas disjuntas de enlace hasta un número máximo de intentos. Esto último permite la generación de más rutas alternativas.

La principal idea de AntOR-UDLR es sustituir los mensajes de notificación de fallo de enlace por mensajes unicast que se envían al predecesor del nodo que informa del fallo del enlace hasta llegar a la fuente de la sesión de datos, ya que en AntOR-DLR se envían en modo difusión (broadcast). La utilización de mensajes unicast hace que se pierdan menos mensajes, porque antes de transmitir se comprueba si está disponible el medio a través del cual se quieren enviar, hecho que no sucede cuando se envía en modo broadcast. Este nuevo protocolo pretende reducir el tráfico de la red, evitando que la información transmitida llegue innecesariamente a nodos que no necesitan procesarla.

AntOR-v2 y HACOR son las dos variantes más evolucionadas, proporcionando nuevas técnicas de optimización como almacenamiento de paquetes de control y gestión de rutas obsoletas, así como diferentes gestiones de fallos de enlace y de exploración de rutas. La principal diferencia entre AntOR-v2 y HACOR está en el proceso de exploración de rutas y radica en la técnica empleada. En AntOR-v2 las hormigas proactivas son enviadas al vecino a 1-salto con mejor valor de feromona. En cambio, en HACOR el proceso proactivo se apoya en la implementación algorítmica S-ACO, que constituye el punto de partida del funcionamiento de los algoritmos ACO. HACOR presenta además nuevas técnicas de neutralización de fallos de enlace.

PAntOR es una versión paralelizada de AntOR que hace uso de arquitecturas multiprocesador de programación de grano grueso basadas en un sistema de memoria compartida por medio de la estandarización Posix Thread, que permite ejecutar tareas en paralelo usando hilos, siendo aplicable esta paralelización en la fase de establecimiento de ruta, en el proceso de reparación local de rutas y en la notificación de fallos de enlace. PAntOR-MI es una variante multi-interfaz, que paraleliza el envío de mensajes de difusión por interfaz a través de hilos.

Finalmente, se han realizado diversas simulaciones en NS-3 con el objetivo de validar las propuestas anteriores. Los resultados de la simulación demuestran que: i) el protocolo

AntOR-DLR presenta mejor throughput y mejor ratio de entrega de paquetes de datos que su predecesor AntHocNet así como un pequeño incremento del retardo medio extremo a extremo y de la sobrecarga, si bien estas dos últimas métricas tienden a igualarse en redes densas; ii) el protocolo AntOR-DLR mejora a AntOR-DNR; iii) el protocolo AntOR-RDLR mejora su predecesor AntOR-DLR; iv) el protocolo AntOR-UDLR también mejora a su predecesor AntOR-DLR en todas las métricas a excepción de la sobrecarga, en cuyo caso ambos protocolos presentan valores similares; v) el protocolo AntOR-v2 mejora a AODV en todas las métricas analizadas, a excepción de la sobrecarga que es ligeramente superior, diferencia ésta que se hace imperceptible en redes densas; vi) el protocolo HACOR mejora a AODV y OLSR en el throughput y en el ratio de entrega de paquetes de datos, presentando valores intermedios respecto a estos dos protocolos en las otras métricas consideradas; vii) el protocolo PAntOR mejora a AntOR-DNR; viii) el protocolo PAntOR-MI mejora a su predecesor en entornos muy dinámicos; y, ix) todos los protocolos especificados se comportan de forma estable en todas las simulaciones realizadas.

Los anteriores resultados permiten concluir que la familia de protocolos secuenciales especificada (del que HACOR es su máximo exponente) mejora la escalabilidad de su predecesor AntHocNet, protocolo que además se presupone mejor para la mayoría de los ambientes que los estándares de encaminamiento reactivo (AODV) y proactivo (OLSR), y que las aproximaciones paralelas consideradas para este tipo de algoritmos de encaminamiento ACO pueden mejorar aún más las prestaciones de esta familia.

16.1 Trabajos Futuros

Aunque los protocolos de encaminamiento especificados ofrecen una solución atractiva para el diseño de una red móvil ad hoc, muchas son las cuestiones que pueden plantearse. Las principales líneas de investigación que se derivan del presente trabajo son:

- **Estudio de posibles modificaciones en AntOR**. Sería interesante analizar otras métricas en el proceso de exploración de rutas (retardo extremo-a-extremo, combinación de retardo extremo-a-extremo con el número de saltos, etc.); aplicar algún proceso de evaporación de feromona (AntOR no lo contempla); utilizar rutas de zonas disjuntas (*Zone-Disjoint Route* [AEOP10]) que, si bien son más restrictivas e independientes que las rutas de enlace (o nodo) disjunto, toleran mejor los fallos de enlace; actualizar las rutas disjuntas en la fase de exploración y descubrimiento de ruta con objeto de reducir la latencia y la sobrecarga, etc.

- **Diseño de nuevas técnicas de paralelización**. Sería muy interesante conseguir implementaciones paralelas más eficientes. En P-AntOR el envío se hace a todos los vecinos utilizando un hilo por cada uno de ellos, lo que origina una alta sobrecarga, llegando incluso a colapsar los dispositivos. Se podría restringir este envío a un número limitado de vecinos elegidos de alguna forma (aleatoriamente, por ejemplo). Asimismo, se podría utilizar otra técnica de paralelización distinta de la memoria compartida mediante Posix Thread y hacer alguna comparativa. En PAntOR-MI se considera que los caminos que se crean entre pares de nodos origen/destino tienen en cuenta la dirección principal de cada nodo, de tal forma que si un nodo recibe el mismo paquete de control, sólo tiene en cuenta la dirección IP principal asociada a ese nodo. Se podría considerar asociar rutas a través de nodos intermedios utilizando otros interfaces que no tengan asociados la dirección principal. Con esta posibilidad se crearían un mayor número de rutas en el proceso de establecimiento de ruta.

- **Extensión segura de los protocolos especificados**. Todos los protocolos de encaminamiento ACO para redes móviles ad hoc están diseñados sin tener en cuenta el comportamiento malicioso de alguno de los nodos, lo que puede ser aprovechado para vulnerar la seguridad de la red. AntOR pertenece a este grupo de protocolos donde los atacantes pueden alterar su comportamiento, de ahí la necesidad de una extensión de este protocolo, al igual que sucede en los protocolos de encaminamiento tradicionales. En este sentido sería conveniente ver la aplicabilidad de algunas de las extensiones de seguridad más comunes desarrolladas para éstos como, por ejemplo, *Coded-Optimized Link State Routing* (COD-OLSR) [GVGMRCSO11], extensión segura de OLSR al que añade una ligera sobrecarga que apenas afecta al rendimiento y que es una interesante alternativa para proveer integridad en OLSR frente a los mecanismos clásicos que hacen de criptografía, más complejos y con una gran sobrecarga.

- **Aplicación de los protocolos de encaminamiento ACO desarrollados para redes móviles ad hoc en otros campos**. Un área de inmediata aplicación serían las redes de sensores [SDCF11] dada la similitud existente entre ambos tipos de redes. Otro campo de aplicación podría ser la robótica [DDCPG11] donde frecuentemente se analiza el uso de interacciones locales simples para resolver tareas complejas como, por ejemplo, la navegación en entornos cerrados.

Part III

Papers Related to This Thesis

Appendix A

List of Publications

- Luis Javier García Villalba, Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco. Bioinspired Routing Protocol for Mobile Ad Hoc Networks. *IET Communications*, 4(18):2187-2195, December 2010 [GVRCSO10].

- Luis Javier García Villalba, Julián García Matesanz, Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco. Secure Extension to the Optimised Link State Routing Protocol. *IET Information Security* 5(3):163-169, September 2011 [GVGMRCSO11].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba. An Extension Proposal of AntOR for Parallel Computing. Proceedings of the Third International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2011), Amman, Jordan, October 10-13, 2011. Journal of Ubiquitous Systems & Pervasive Networks 3(2):67-72, October 2011 [RCSOGV11].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba, Tai-Hoon Kim. A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks. Part II of Proceedings of the 2011 International Conference on Multimedia, Computer Graphics and Broadcasting (MulGraB 2011), Jeju Island, Korea, December 8-10, 2011. *Communications in Computer and Information Science* 263:300-304, December 2011 [RCSOGVK11a].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba, Tai-Hoon Kim. Comparing AntOR-Disjoint Node Routing Protocol with Its Parallel Extension. Part II of the Proceedings of the 2011 International Conference on Multimedia, Computer Graphics and Broadcasting (MulGraB 2011), Jeju Island, Korea, December 8-10, 2011. *Communications in Computer and Information Science* 263:305-309, December 2011 [RCSOGVK11b].

- Delfín Rupérez Cañas, Luis Javier García Villalba. Immune Systems for ACO-Based Routing Optimization. Book of Abstracts of the 11th International Conference on Artificial Immune Systems (ICARIS 2012), Taormina, Italy, August 28-31, 2012 [RCGV12].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villaba. ANTOR-UDLR: Aproximación Unicast de un Protocolo de Encaminamiento para Redes Móviles Ad Hoc. Actas del XXVII Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2012), Elche, Alicante, Spain, September 12-14, 2012 [RCSOGV12a].

- Luis Javier García Villalba, Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Tai-Hoon Kim. Multiple Interface Parallel Approach of Bioinspired Routing Protocol for Mobile Ad Hoc Networks. *International Journal of Distributed Sensor Networks*, 2012 (ID 532572):1-5, October 2012. [GVRCSOK12a].

- Luis Javier García Villalba, Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Tai-Hoon Kim. Restrictive Disjoint-Link-Based Bioinspired Routing Protocol for Mobile Ad Hoc Networks. *International Journal of Distributed Sensor Networks*, 2012(ID 956146):1-5, October 2012. [GVRCSOK12b].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villaba. Technique to Neutralize Link Failures for an ACO-Based Routing Algorithm. Proceedings of the 13th Edition of the Ibero-American Conference on Artificial Intelligence (IBERAMIA 2012), Cartagena de Indias, Colombia, November 13-16, 2012. *Lecture Notes in Artificial Intelligence* 7637:251-260, November 2012 [RCSOGV12b].

- Luis Javier García Villalba, Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco. Parallel Approach of a Bioinspired Routing Protocol for MANETs. *International Journal of Ad Hoc and Ubiquitous Computing* 12(3):141-146, March 2013 [GVRCSO13].

- Delfín Rupérez Cañas, Luis Javier García Villalba, Ana Lucila Sandoval Orozco, Tai-Hoon Kim. Adaptive Routing Protocol for Mobile Ad Hoc Networks. *Computing*, 2013:1-11, March 2013 [RCGVSOK13].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba. An Ant-Based Adaptive Distributed Routing Protocol for Mobile Ad Hoc Networks. Proceedings of the 6th International Conference on Information Technology (ICIT 2013). Amman, Jordan, May 8-10, 2013 [RCSOGV13a].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba, Pil-Seung Hong. HACOR: Hybrid ACO Routing Protocol for Mobile Ad Hoc Networks. *International Journal of Distributed Sensor Networks*, 2013:1-7, May 2013 [RCSOGVH13].

- Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba. Routing Techniques Based on Swarm Intelligence. Proceedings of the 7th International Conference on Intelligent Systems and Knowledge Engineering (ISKE 2012), Beijing, China, December 13-15, 2012. *Advances in Intelligent Systems and Computing* (in press) [RCSOGV13b].

Published in IET Communications
Received on 23rd December 2009
Revised on 30th May 2010
doi: 10.1049/iet-com.2009.0826

ISSN 1751-8628

Bio-inspired routing protocol for mobile *ad hoc* networks

L.J.G. Villalba D.R. Cañas A.L.S. Orozco

Grupo de Análisis, Seguridad y Sistemas (GASS), Departamento de Ingeniería del Software e Inteligencia Artificial, Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM), Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain
E-mail: javiergv@fdi.ucm.es

Abstract: A mobile *ad hoc* network (MANET) is a non-centralised, multihop, wireless network that lacks a common infrastructure. It therefore needs self-organisation. The MANETs are autonomous, adaptive and dynamic. In MANETs, multihop routing allows multiple routes between the source and destination to be established. This study speaks about a bio-inspired routing protocol for these networks based on AntHocNet. The design for the protocol lies in a heuristic, based on swarm intelligence, which takes into account the limited resources and highly dynamic environment, as well as the restriction on the exchange of routing information. So, the key aspects of the proposed protocol are the disjoint-link and disjoint-node routes, separation between the regular pheromone and the virtual pheromone in the diffusion process and the exploration of routes, taking into consideration the number of hops in the best routes which the authors have previously found out. The simulation results show that the protocol has lower overhead and higher delivered packet ratio than AntHocNet. Likewise, these results indicate that the routing satisfies multiple and independent quality of service constraints and can deal with faults, which provides better load balancing.

1 Introduction

A mobile *ad hoc* network (MANET) [1] is a group of mobile nodes which operate themselves through wireless links. In contrast with the conventional networks, a MANET does not need an infrastructure, since nodes rely on each other to operate themselves, forming what is called multihop communication. Such networks have more problems and disadvantages than a conventional network.

The topology of mobile networks may change quickly and without warning. As a result, errors in the transmission are unpredictable and a lack of security is common, both in nodes and links. We must add that the nodes have limited resources, since an *ad hoc* network is usually made by battery-powered devices with low capacity, memory and so on.

Basically, routing protocols based in MANETs are classified into three categories: proactive, reactive and hybrid. Proactive routing protocols often need to exchange control packets among mobile nodes and continuously update their routing tables. Each node must maintain the state of the network in real time. This causes high overhead congestion of the network, which requires lots of memory. The advantage of proactive protocols is that nodes have correct and updated information. So, when we need a path, we can find it directly in memory and establish links quickly. These protocols are intended to reduce broadcasting frequency while maintaining the correct information for the routing table. Reactive routing protocols only seek a route to the destination when it is needed. The advantage of these protocols is that the routing tables located in memory are not continuously updated. On the other hand, they have the disadvantage that they cannot establish connections in real time. The aim of these protocols is to save time in the route discovery process, since the reactive protocol is designed to reduce the latency which is critical in this kind of protocols. It also aims to avoid the maintenance of routes to produce long delay.

Hybrids are derived from a mixture of these two protocols, and for this reason, they share some of their advantages. This paper, which shows an innovative routing algorithm, is organised as follows. In Section 2, we present related

works, where we expose some of the bio-inspired protocols based specifically on the ant behaviour, ant colony optimisation (ACO), for *ad hoc* networks. In Section 3, we expose a base protocol which supports our approach, describing and analysing it to make a comparative study. In Section 4, we propose our protocol, explaining the features in detail. Then, in Section 5, the most relevant results are shown. Finally, the paper concludes in Section 6 with overall conclusions, observations and potential advancements for further investigations.

2 Related works

MANETs have special characteristics that must be taken into account when a routing protocol is implemented. There are many solutions (RFC 3626 [2], RFC 3561 [3], RFC 4728 [4], RFC 3684 [5], etc.). All these protocols have valid solutions, but they usually have a specific topology and characteristics of certain scenarios as a design basis. They are not always particularly suitable if there are drastic changes in the dynamic topology of the network. There is a group of algorithms or routing protocols called bio-inspired, whose essential characteristic consists of being adaptive, which is especially noteworthy in this kind of network. The concept of swarm intelligence [6] is specifically referred to in the literature. It is based on the application of social behaviour of insects and other animals to solve the problems. The ACO [7] algorithm is the starting point of these algorithms. The ACO algorithms are based on the collective behaviour of ants in search of food to bring back to the nest. Various tasks are performed by proposals of ACO routing, in which proactive, reactive and hybrid protocols are found.

Since proactive ACO routing protocols are included, probabilistic emergent routing algorithm [8] has a low delivered data packet ratio in scenarios with high mobility; it has a high overhead caused by control messages being sent several times in broadcast mode. Another protocol is ant routing algorithm for MANETs [9] which, according to the authors, reduces the overhead of discovery and maintenance of the routes; but, they do not discuss how they control the generation of control messages in a dynamic environment. However, it has the common characteristic of achieving a low latency in the route discovery process, with the information of the routing table receiving correct updates. We also mention reactive protocols called ant-colony-based routing algorithm (ARA) [10]. This approach made use of the process of flooding to update pheromone tables in all nodes. This process has greater scope in the transmission of packets than a simple broadcast, but leads to high overhead. ARA is not scalable and does not detect loops. Ant colony-based multipath quality of service (QoS)-aware routing [11] protocol is robust and can withstand better QoS, but is similar to reactive protocol, which has a high latency in the discovery of routes. As hybrid ACO routing protocols are included, ant *ad hoc* on-demand distance vector [12], in which the latency of route discovery is reduced, because the process of route discovery is reduced. Hybrid ACO routing algorithm for MANET [13] is a highly scalable protocol, with the disadvantage that, when the number of nodes is low, the continuous movement of the peripheral nodes incites to discover new routes causing more delay than other hybrid protocols, and AntHocNet [14–16] protocol is based on the approach made in this article. This protocol does not take into account disjoint-link/node routes and has a high overhead in the process of exploring new routes. The disadvantage of the previously reviewed protocols is not using disjoint routes, whether link or node.

3 AntHocNet

AntHocNet [15, 16] is a hybrid ACO routing algorithm. Data from 2004 and 2005 have had numerous extensions and a great impact, being a pioneer algorithm in this field. This protocol is applied to multipath and dynamic networks, that is, creating multiple paths to transmit data from source to destination in the same data session. AntHocNet follows a structure similar to AntNet-fast ant (AntNet-FA) [17], but it differs from AntNet-FA given that topologies of static networks are applied and convergence is slow. So, what all ants have to do is choose the path. AntHocNet, meanwhile, takes into account the dynamic topology and other characteristics of *ad hoc* networks. When the network topology changes, then it must be restored quickly and this is achieved through a new route discovery process. If multiple resources are used to accelerate this process, the exchange of information is enhanced. This can cause the network to collapse. Therefore we have a problem that if we do not want to overload the network, we increase the convergence time (time required for the network to become stable before a link failure) of the ACO algorithm, and if we want to reduce the convergence time, we overload the network. The previous problem means that AntNet-FA algorithm does not directly apply to MANETs, so it needs a modification to accelerate its convergence time without overloading the network. In this way, AntHocNet emerges as a reactive, adaptive, multipath and proactive algorithm (hybrid). It is reactive because it has agents operating on-demand to set up routes to destinations.

- It is proactive because it has agents collecting information and these agents can discover new routes which serve as alternatives if a link fails.

- It is a multipath because it provides different routes to send information to the destination.

- It is adaptive because it adapts to traffic conditions and networks.

In the operation of AntHocNet, the following stages or phases are distinguished:

doi: 10.1049/iet-com.2009.0826

• *Routing information setup:* The source node sends reactive agents to discover the first available route to the destination.

• *Data routing:* Data are sent through the nodes to the destination using the route information and can use a multihop technique, which involves sending data through intermediate nodes. These nodes act as routers.

• *Path maintenance and exploration:* Information about existing routes is proactively updated and the discovery of new ones is possible.

• *Management of link failures:* Management failures occur when a node is outside the scope of the network or does not receive control messages which are responsible for informing a node of its closest neighbours (who are one hop), and so on. This phase deals with such failures.

4 AntHocNet-based improved routing (AntOR): routing approach

The proposed protocol, AntHocNet-based improved routing (AntOR), is a hybrid ACO routing protocol that takes Ducatalle algorithm [18] as its starting point, and demonstrates the following qualities and abilities, which are different from AntHocNet:

• disjoint-link and disjoint-node protocol;

• separation between the pheromones in the diffusion process;

• use distance metric in path exploration.

4.1 Disjoint-link/node protocol

In such protocols there are two kinds of routes: disjoint-node and disjoint-link. The first corresponds to routes in which nodes are not shared and the latter refers to routes in which links are not shared. Every disjoint-node is also a disjoint-link, but not vice versa.

Both types of disjoint routes have the following advantages:

1. A failure in one node only affects a path, not the entire network.

2. Load balancing is better because there are not repeated routes on the disjoint property.

However, the use of such routes does also have its disadvantages, for example

1. More resources are needed because they do not share links and nodes.

2. These routes are more difficult to detect, because we limit the nodes that can be explored.

Our approach does not use disjoint-partial routes as in [19]. Disjoint-link and disjoint-node are different from the disjoint-partial because they may have both nodes and links in common. Disjoint-partial routes are less restrictive, allowing computing of more routes, providing better tolerance to link failures as a result of quick and efficient recovery from broken routes. Also, in general, they are easier to detect. But they have the disadvantage of using some intermediate node of the main route (hybrid approach between disjoint-link and disjoint-node), so whether an intermediate node fails a new route has to be discovered. Therefore the routes which are only disjoint-link or disjoint-node prevent better link failures because they have alternatives so that nodes and links are not repeated. This is one of the reasons not to use disjoint-partial routes. Next, we will show how to calculate disjoint-node and disjoint-link with regard to AntOR algorithm.

4.1.1 Disjoint-node routes: As mentioned above, a disjoint-node route is one in which nodes are not shared in the same data session. Fig. 1 shows an example of disjoint-node routes. The procedure for calculating disjoint-node route is different from how we calculate disjoint-link. In the case of disjoint-node, the intermediate nodes which have been visited are marked, so we do not repeat them. This table shows information regarding the previously visited nodes, which is stored 'locally'. This aims to perform a proactive process for the exploration of new routes. In Fig. 2 a flowchart shows the functionality of this method.

4.1.2 Disjoint-link routes: As previously mentioned, a disjoint-link route is one in which no links are shared on a same data session. Fig. 3 shows an example of disjoint-link route, which recognises the difference between disjoint-link route and non-disjoint-link route. The basic idea for finding and representing disjoint-link routes is to mark each disjoint-link with a label indicating what the source of the data session is. In Fig. 4, we present a diagram showing how this process functions. We can see that the mechanism is very simple.

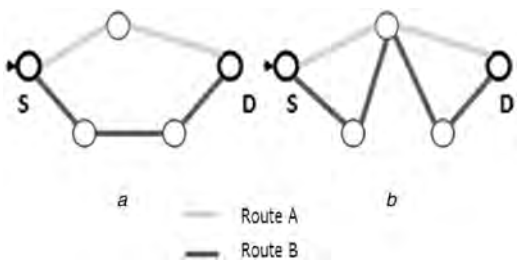


Figure 1 *Node-disjoint routes (a) against no node-disjoint routes (b)*

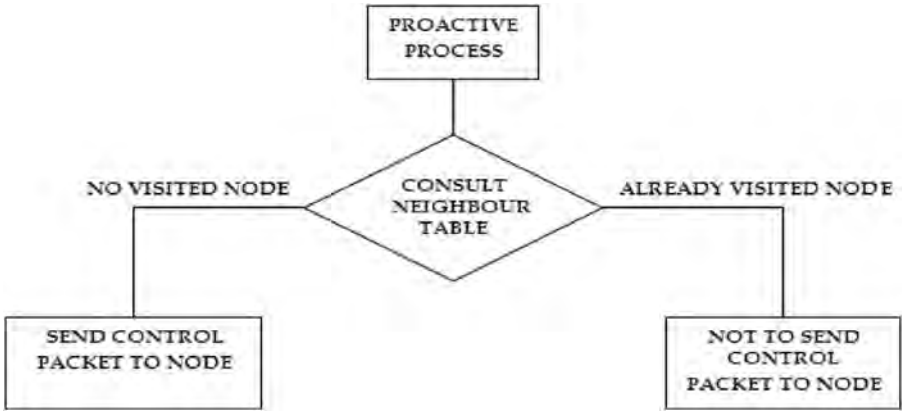


Figure 2 *Operation of node-disjoint route*

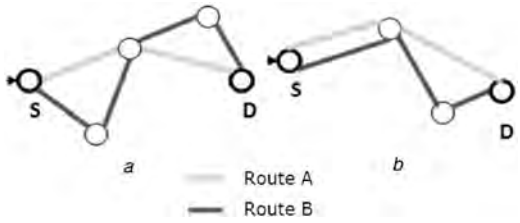


Figure 3 *Link-disjoint routes (a) against no link-disjoint routes (b)*

When two or more data sessions sharing links in its information retransmit, the procedure to calculate disjoint-link routes is similar: each link which is visited is marked according to the session that it belongs to. As a result, there will be no conflict or problems since the sessions do not rely on each other.

4.2 Separation between the pheromones in the diffusion process

In Ducatelle's approach [18], the same route can have both regular pheromone values and pheromone virtual values simultaneously. Regular pheromone values are used to determine the routes through which data travels. On the other hand, virtual pheromone values indicate the routes that can possibly be 'good' to relay the data.

In AntOR, a route cannot have both a regular pheromone value and a virtual pheromone simultaneously; this technique improves the efficiency of the algorithm. To carry out this separation, the equation (1) of Ducatelle's route exploration is changed [18].

$$P_{in}^{d} = \frac{\max(\tau_{in}^{d}, k_{in}^{d})^{\beta_2}}{\sum_{j \in N_i^d} \max(\tau_{ij}^{d}, k_{ij}^{d})^{\beta_2}} \quad (1)$$

where P_{in}^{d} is the likelihood that node i chooses the next hop n with destination node d; $\max(a, b)$ function returns the maximum of two values a and b; τ_{in}^{d} corresponds to the regular pheromone value; k_{in}^{d} corresponds to the virtual pheromone value; β_2 is a protocol configuration parameter. It is used in the proactive process for new routes exploration. When this parameter has a low value, all alternative routes to a destination can be chosen with equal probability N_i^d. This is a one-hop neighbour of node i.

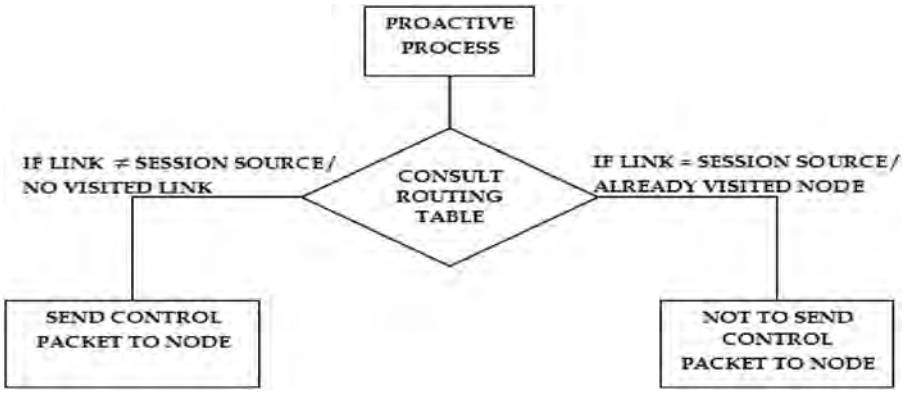


Figure 4 *Operation of link-disjoint route*

doi: 10.1049/iet-com.2009.0826

The result of this modification of (1) is shown in the following one.

$$P_{in}^{d} = \frac{(\psi_{in}^{d})^{\beta_2}}{\sum_{j \in N_i^d} (\psi_{ij}^{d})^{\beta_2}}, \quad \psi \in \begin{cases} k & \text{virtual} \\ \tau & \text{regular} \end{cases} \tag{2}$$

where ψ corresponds to a regular or virtual pheromone value, but never corresponds to both simultaneously.

Then, an example to compare the difference of (1) and (2) is shown. There are four nodes (A, B, C and D) in the scenario. As node A wants to send data to the destination node D, node A needs to choose the most appropriate route. Fig. 5 shows the scenario from the point of view of Ducatelle equation or (1), where the two routes from A have regular and virtual pheromones. Therefore to calculate the probability of choosing the alternative route 1, the following equation is used

$$P_{\mathrm{AC}}^{\mathrm{D}} = \frac{\max(\tau_{\mathrm{AC}}^{\mathrm{D}}, k_{\mathrm{AC}}^{\mathrm{D}})^{\beta_2}}{\max(\tau_{\mathrm{AB}}^{\mathrm{D}}, k_{\mathrm{AB}}^{\mathrm{D}})^{\beta_2} + \max(\tau_{\mathrm{AC}}^{\mathrm{D}}, k_{\mathrm{AC}}^{\mathrm{D}})^{\beta_2}} \tag{3}$$

On the other hand, Fig. 6 shows the same scenario but from the perspective of AntOR. In this case, the alternative route 1 has only regular pheromone and the route 2 has only virtual pheromone. Thus, the probability of choosing the alternative route 1 is given by next equation

$$P_{\mathrm{AC}}^{\mathrm{D}} = \frac{(\tau_{\mathrm{AC}}^{\mathrm{D}})^{\beta_2}}{(\tau_{\mathrm{AC}}^{\mathrm{D}})^{\beta_2} + (k_{\mathrm{AB}}^{\mathrm{D}})^{\beta_2}} \tag{4}$$

These two equations (3) and (4) use the pheromone in a different way to calculate the alternative routes. In the case of (3) a route can have regular and virtual pheromones simultaneously. In the case of (4) the same route has regular or virtual pheromone, but not both at the same time.

4.3 Use distance metric in path exploration

Our approach takes into account the number of hops for the routes which have been found to be the best. There is a hop limit on the nodes. This hop limit is established according to previously calculated routes that have a smaller distance in hop number. Fig. 7 shows an example of how this mechanism which we have mentioned in our approach works.

Once you have established the main route, sending proactive control packets to explore new routes can be

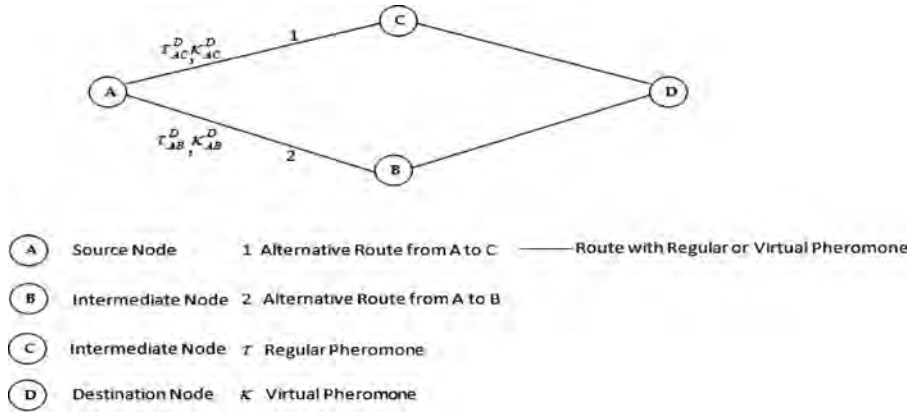


Figure 5 *Illustrative example of (1)*

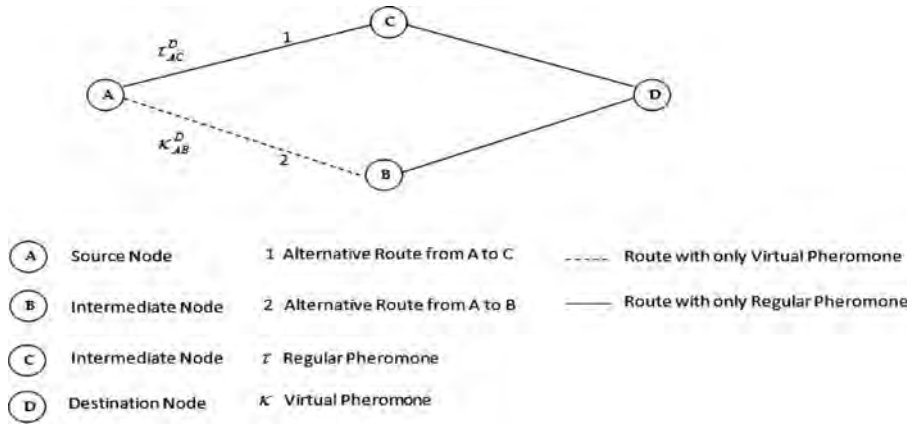


Figure 6 *Illustrative example of (2)*

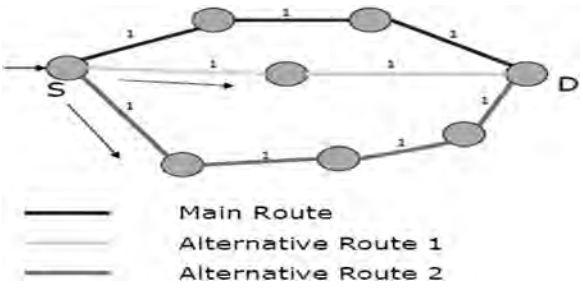


Figure 7 *Selection of the distance metric*

chosen from two alternatives: (a) alternative route 1 and (b) alternate route 2. In this proactive process, route 1 will be chosen because it has a hop count less than the main route because the main route consists of three hops. Thus, the control messages that they choose route 2 will never reach their destination because they are scheduled at most to visit three nodes.

5 Illustrative results

In this section, we show the most relevant results from what we discovered in the simulation.

5.1 Parameter setting

To evaluate the behaviour of the protocol network simulator 3 (NS-3) has been used [20]. We compared AntOR-DLR (disjoint-link route version) with AntHocNet. So, we have used common parameters for both the protocols, as is shown in Table 1. Table 2 shows the internal characteristics of both protocols that are compared. Finally, in Table 3 we indicate the parameters of the scenario, where you can see mobility as an important characteristic: random waypoint (RWP). We also used a highly dynamic scenario (maximum speed of nodes is 10 m/s) because it is attempting to analyse the worst case. Next, we show the obtained results by comparing both protocols using the same scenario, according to different performance parameters.

5.2 Performance metrics

The most important performance metrics that were used to assess this approach are

Table 1 General features of the simulations

number of nodes: between 20 and 100
dimensions of area: 1400 × 1400
transmission range (open area): 300 m
physical layer: configured for IEEE 802.11b
sent data ratio: constant WiFi-6 mps
time simulation: 30 s
number of trials: 3

Table 2 Parameters of configuration

$\gamma = 0.7$
$\alpha = 0.7$
$\beta_1 = 20$
$\beta_2 = 2$
$\beta_3 = 20$
maximum number of destinations in the HELLO message: 10
HELLO emission interval: 1 s
PFA emission interval: 2 s
maximum number of retry for restoring the route: 5
RFA emission interval: 5 s
maximum number of broadcast allowed by RRFA message: 2

Table 3 Parameters of stage

beginning of time CBR client: 0 s
ending of time CBR client: 30 s
beginning of time CBR server: 0 s
ending of time CBR server: 30 s
number of data sessions: 4
position model: list
mobility model: RWP
velocity: minimum 0 and maximum 10 m/s
pause time: 5 m/s

- *Throughput:* Volume of work or information flowing through a system. It is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.

- *Delivered data packet ratio:* Relationship between number of packets sent and the number of packets delivered successfully.

- *Average end-to-end delay:* Measure of accumulative effectiveness of experienced delays by packets going from source to destination.

- *Overhead in number of packets:* Relationship between the total numbers of transmitted control packets by the nodes of network and the number of delivered data packets to their destinations.

- *Overhead in number of bytes:* Relationship between the total number of transmitted control bytes and delivered data bytes.

doi: 10.1049/iet-com.2009.0826

5.3 Throughput

Fig. 8 shows that throughput in AntOR in extreme situations (space networks or very dense) is higher than AntHocNet's throughput. In average situations, the values are made equal. Similarly, a robustness of the protocol is shown since the curve displays a greater stability against variations in network conditions.

5.4 Delivered packet ratio

Fig. 9 shows that delivered packet ratio is similar to throughput seen previously, but with the use another scale. This is because the delivered packets influence in both the metrics, and we can also see how AntOR is scalable, obtaining a good ratio with 100 nodes.

5.5 Average end-to-end delay

Fig. 10 shows that average end-to-end delay of AntOR in sparse network is superior to finding in AntHocNet. However, the value of this delay is not very high. For denser networks, AntOR is practically made equal to AntHocNet. Although the latter has better values of delays in sparse networks, it seems to be more unstable against increasing nodes than AntOR.

5.6 Overhead in number of packets

Fig. 11 shows that the overhead in the number of packets of AntOR is lower than AntHocNet, which proves that the changes made are correct. The disjoint-link capacity means less proactive forward ant (PFA) control messages are sent. With better tolerance of link failures, thanks to the existence of alternative routes, fewer repair route forward ant (RRFA) control message in the route repair process needs to be sent.

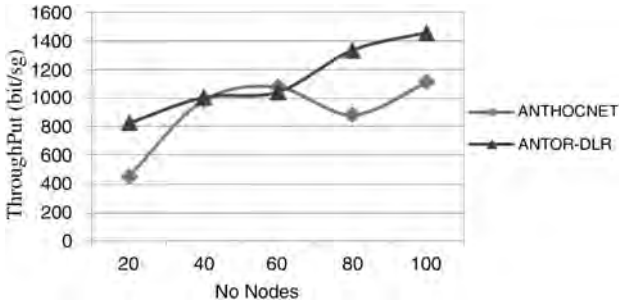


Figure 8 *Throughput*

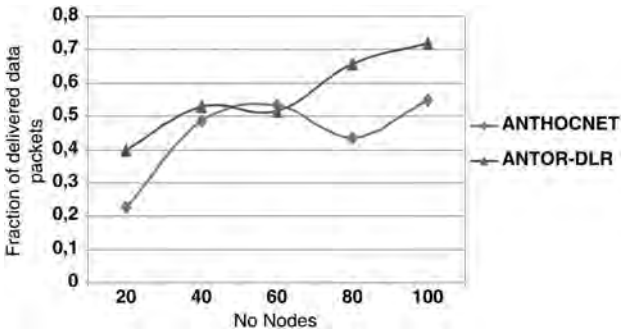


Figure 9 *Delivered data packet ratio*

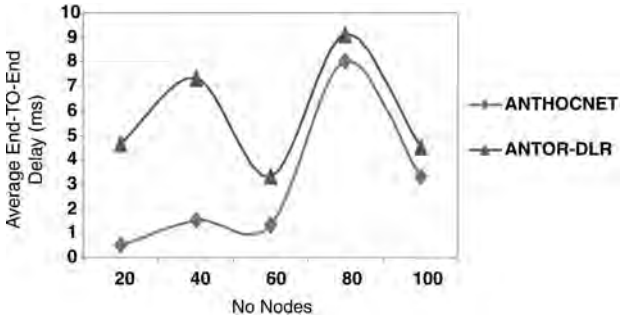


Figure 10 *Average end-to-end delays*

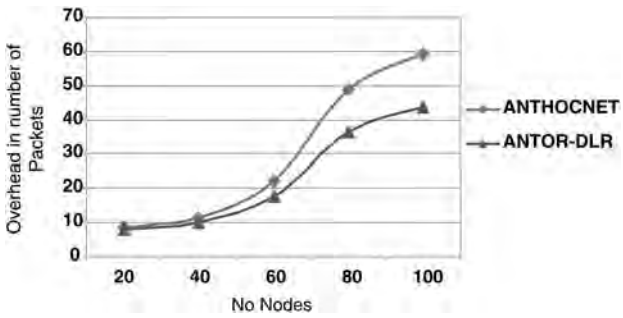


Figure 11 *Overhead in number of packets*

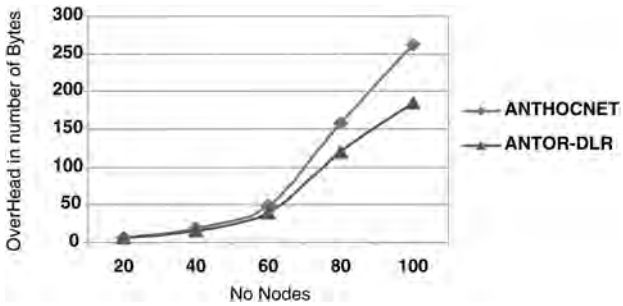


Figure 12 *Overhead in number of bytes*

5.7 Overhead in number of bytes

Fig. 12 appreciated that in the case of sparse networks, the overhead in the number of bytes (which is a more representative parameter) is similar in both the protocols, while in the case of denser networks, there is a reduced overhead than with AntHocNet.

6 Conclusions

We have presented a routing protocol for MANETs called AntOR that is classified as a hybrid ACO routing protocol (based on the algorithm of ant colony) and can be

considered as a variant of the AntHocNet protocol, which improves the performance in important parameters such as delivered packet ratio, the overhead in the number of packets and the overhead in the number of bytes. The protocol is stable in the carried out simulations, which suggested its scalability. Even more outstanding characteristics of this variant approach of AntHocNet can be noticed. Several are mentioned below:

- use disjoint-link/node routes;

- the separation between the pheromones in the diffusion process;

- the new route exploration process takes into account the hop number of the best routes found previously.

These latest characteristics and qualities have as a goal, the reduction of the overhead in the number of bytes and the increase of delivered packet ratio of AntHocNet, whose aspects have been achieved with this approach.

There are several possible lines of work which could be developed further. Among them are the following:

- Specify new simulation scenarios to validate the obtained result.

- Analyse other performance metrics such as jitter.

- Implement the disjoint-node version (it has implemented disjoint-link version).

- Make a comparison with more reference protocols.

- Implement disjoint-partial route version by comparing this approach with the proposed approach.

7 Acknowledgments

This work was supported by the Ministerio de Ciencia e Innovación (MICINN) through Project TEC2007-67129/TCM and the Ministerio de Industria, Turismo y Comercio (MITyC) through the Projects AVANZA I + D TSI-020100-2008-365 and TSI-020100-2009-374.

8 References

[1] IETF Mobile ad-hoc networks (MANET) working group: http://www.ietf.org/html.charters/manet-charter.html

[2] CLAUSEN T., JACQUET P.: 'Optimized link state routing protocol (OLSR)'. IETF RFC 3626, October 2003, http://www.ietf.org/rfc/rfc3626.txt

[3] PERKINS C.E., BELDING-ROYER E.M., DAS S.: '*Ad hoc* on-demand distance vector (AODV) routing'. RFC3561, July 2003, http://tools.ietf.org/html/rfc3561

[4] JONSON D., HU MALTZ D.: 'The dynamic source routing protocol (DSR) for mobile *ad hoc* networks for IPv4'. RFC 4728, February 2007

[5] OGIER R., TEMPLIN F., LEWIS M.: 'Topology dissemination based on reverse path forwarding (TBRPF)'. IETF RFC 3684, February 2004, http://www.ietf.org/rfc/rfc3684.txt

[6] KENNEDY J.: 'Swarm intelligence' (Morgan Kaufmann Publishers, 2001)

[7] DORIGO M., STÜTZLE T.: 'Ant colony optimization' (The MIT Press, 2004)

[8] BARAS J.S., MEHTA H.: 'A probabilistic emergent routing algorithm for mobile *ad hoc* networks'. Modeling and Optimization in Mobile *Ad Hoc* Wreless Networks (WiOpt'03), March 2003

[9] HOSSEIN O., SAADAWI T.: 'Ant routing algorithm for mobile *ad hoc* networks (ARAMA)'. Proc. 22nd IEEE Int. Performance, Computing, and Communications Conf., Phoenix, Arizona, USA, April 2003, pp. 281–290

[10] GÜNES M., SORGES U., BOUAZIZI I.: 'ARA – the ant-colony based routing algorithm for MANETs'. Proc. ICPP Int. Workshop on *Ad Hoc* Networks (IWAHN), 2002

[11] LIU L., FENG G.: 'A novel ant colony based QoS-aware routing algorithm for MANETs' (Springer, Berlin, Heidelberg, 2005), *ICNC*, **2005**, *LNCS*, **3612**, pp. 457–466

[12] MARWAHA S., THAM C.K., SRINAVASAN D.: 'Mobile agents based routing protocol for mobile *ad hoc* networks'. IEEE Global Telecommunications Conf. (GLOBECOM'02), Taipei, Taiwan, 2002

[13] WANG J., OSAGIE E., THULASIRAMAN P., THULASIRAM R.K.: 'HOPNET: a hybrid ant colony optimization routing algorithm for mobile *ad hoc* network', *Ad Hoc Netw.*, 2009, **7**, (4), pp. 690–705

[14] DI CARO G.: 'Ant colony optimization and its application to adaptive routing in telecommunication networks'. PhD thesis in Applied Sciences, Polytechnic School, Université Libre de Bruxelles, Brussels, Belgium, 2004

[15] DI CARO G.A., DUCATELLE F., GAMBARDELLA L.M.: 'AntHocNet: an adaptive nature-inspired algorithm for routing in mobile *ad hoc* networks', *Eur. Trans. Telecommun.* (Special Issue on Self-organization in Mobile Networking), 2005, **16**, (5), pp. 443–455

doi: 10.1049/iet-com.2009.0826

[16] DI CARO G., DUCATELLE F., GAMBARDELLA L.M.: 'AntHocNet: an ant-based hybrid routing algorithm for mobile *ad hoc* networks'. Proc. Eight Int. Conf. on Parallel Problem Solving from Nature (PPSN VIII), Birmingham, UK, 18–22 September 2004, (*LNCS*, **3242**)

[17] DI CARO G., DORIGO M.: 'Two at colony algorithms for best-effort routing in datagram networks'. Proc. Tenth IASTED Int. Conf. on Parallel and Distributed Computing and Systems (PDCS'98), 1998, pp. 541–546

[18] DUCATELLE F.: 'Adaptive routing in *ad hoc* wireless multi-hop networks'. PhD thesis, Università della Svizzera Italiana, Istituto Dalle Molle di Studi sull´Intelligenza Artificiale, 2007

[19] ZAFAR H., HARLE D., ANDONOVIC I., KHAWAJA Y.: 'Performance evaluation of shortest multipath source routing scheme', *IET Commun.*, 2009, **3**, (5), pp. 700–713

[20] The NS-3 network simulator: http://www.nsnam.org

Published in IET Information Security
Received on 13th April 2010
Revised on 31st October 2010
doi: 10.1049/iet-ifs.2010.0085

ISSN 1751-8709

Secure extension to the optimised link state routing protocol

L.J. García Villalba[1] *J. Garcia Matesanz*[2] *D. Rupérez Cañas*[1] *A.L. Sandoval Orozco*[1]

[1]*Departamento de Ingeniería del Software e Inteligencia Artificial, Grupo de Análisis, Seguridad y Sistemas (GASS), Universidad Complutense de Madrid*
[2]*Sección Departamental de Sistemas Informáticos y Computación, Grupo de Análisis, Seguridad y Sistemas (GASS), Universidad Complutense de Madrid*
E-mail: javiergv@fdi.ucm.es

Abstract: The design of routing protocols for mobile *ad hoc* networks rarely contemplates, in most cases, hostile environments. Consequently, it is common to add security extensions afterwards. One of the most important routing protocols is the optimised link state routing (OLSR), which in its specification assumes the trust of all nodes in the network, making it vulnerable to different kinds of attacks. This study presents an extension of OLSR, called COD-OLSR, which provides security for OLSR in the case of incorrect message generation attacks which can occur in two forms (identity spoofing and link spoofing). This is one of its main features, which takes into account the current topology of the node sending the message. The behaviour of COD-OLSR against different attackers in a variety of situations is evaluated. The simulation results show that COD-OLSR adds a slight overhead to OLSR and barely affects performance. The results also show that COD-OLSR is an interesting alternative to provide integrity in OLSR compared with classical mechanisms making use of cryptography, which is more complex and has a high overhead.

1 Introduction

Mobile *ad hoc* networks (MANETs) [1] are formed by mobile devices that can communicate with each other without appealing to a preexisting network infrastructure. Most routing protocols for these networks are designed without taking into account the possible malicious behaviour of any of the nodes, something that can be exploited to violate the security of the network. Optimised link state routing (OLSR) [2] belongs to this group of protocols where attackers can alter their behaviour, hence the need for an extension of this protocol. There are several techniques to provide integrity to the OLSR protocol. One of the most widespread is the use of digital signatures for authentication of the OLSR routing messages, of which there are two different variants or approaches: hop-by-hop and end-to-end.

Halfslund *et al.* [3] present a hop-by-hop approximation, where each node signs OLSR packets that are transmitted, discarding the signature of the previous node. This signature only verifies that the node which forwards the message is the same as the one that signs it, but it does not verify the authenticity of the original message. The implemented solutions use shared keys (symmetric) for signature creation and verification. However, they do not mention how the administration/exchange of keys or the initial authentication is carried out. They also propose a method to prevent replay attacks using timestamps, but the overhead increases significantly with this method.

Adjih *et al.* [4] propose schemes to authenticate OLSR messages in an end-to-end approach so that the nodes that receive OLSR messages can authenticate the node which generated the original message. However, replay attacks are still possible. To avoid such attacks another approach has been developed which is based on a distributed timestamp that can verify whether a message is obsolete or not.

Raffo *et al.* [5] design another end-to-end extension where the nodes send OLSR messages as well as an ADVanced Signature message (ADVSIG). Nodes that announce links sign the messages to authenticate the node that originated the message. Mechanisms are established which improve the protection of the protocol against external attacks and from other nodes, although this does significantly add to the overheads.

Raffo *et al.* [6] supply a method that includes the geographical position of the sender node in the control messages as well as an evaluation of the similarity of the links at the same time. This technique has the disadvantage of requiring specific hardware (GPS and directional antennas).

Papadimitratos and Haas [7] provide security in the routing of the link state by using asymmetric primitives. To do this, they assume that each node in the network has a pair of public/private keys, and diffusing in broadcast its public key certificate to the other nodes that are within N hops. These broadcast messages can be periodic or not, depending on changes in the topology of the network, allowing a new node to find out the key to enter the area. Despite using distributed certification the overhead introduced is also important.

Finally, Nait-Abdesselam [8] presents a scheme for detecting and preventing 'Relay' attacks (Wormhole attacks). This technique is based on firstly an identification of the potential links in which the attack occurs, then in distinguishing the links where the Wormhole attack occurs from the ones that

are correct. In order to achieve this, encrypted packets are exchanged between the two neighbours that carried out the attack. This solution is independent of any time synchronisation or location of information, but it has the disadvantage of increasing the overhead because of the exchange of messages for attack detection.

This paper presents a framework that allows us to maintain the integrity of OLSR messages with minimal overhead. It is structured into five sections. In this first section the problem is introduced through displays of related works in which various security extensions to OLSR are discussed. In Section 2, we recall some features of OLSR and we briefly analyse the major attacks it has been exposed to. In Section 3, we specify the security extension, providing the main results of this work. Section 4 contains the most relevant results of the simulation. Finally, Section 5 contains conclusions as well as directions for future research.

1.1 Attacks on OLSR

As its name suggests, the OLSR protocol uses an optimised flooding mechanism to disseminate information to the nodes and to build the network topology. Each node selects a set of neighbour nodes as multipoint relays (MPRs) that are responsible for relaying the control traffic (see Fig. 1) that is diffused in the network. In order to effect this diffusion process, the control message HELLO and TC are used.

HELLO messages allow the neighbouring nodes to be known and can calculate the MPRs. HELLO messages are exchanged periodically only between neighbour nodes, providing several lists of neighbours for each node. One of the lists contains the neighbours that have been picked up by the current node, but have not confirmed the bidirectional link yet. Another list is made up of neighbours who have established a bidirectional communication (symmetrical link). The last list contains neighbour nodes that have been selected by the origin of the HELLO message to act as MPR. In Fig. 2, we show the format of the HELLO messages and their fields.

TC messages are periodically sent through the MPRs. Its purpose is to diffuse topology information to the entire network. A TC message contains the set of bidirectional links between a node and its neighbours. In Fig. 3, we show the TC message format. The list of neighbours is included in this collection of the MPR selector set (MS), that is, those nodes that have chosen to emit the TC message as MPR.

There are other control messages with different functionalities, but we will not consider them, since they are not the main goal of this research.

The main attacks [9, 10], which OLSR is exposed to, are closely related to previous control messages. We distinguish the following kinds of attacks:

- *Incorrect message generation (IMG)*: A node can have bad behaviour in the generation of messages in the following cases: (i) when there is identity spoofing, that is, a node is incorrectly identified as another one, and (ii) when a node announces links with incorrect information in the control messages (link spoofing).

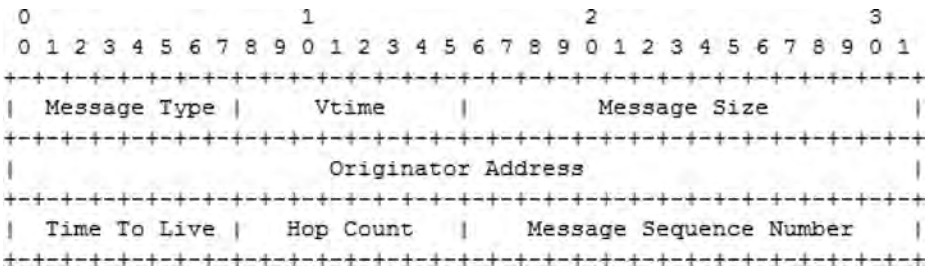


Fig. 1 *OLSR message header format*

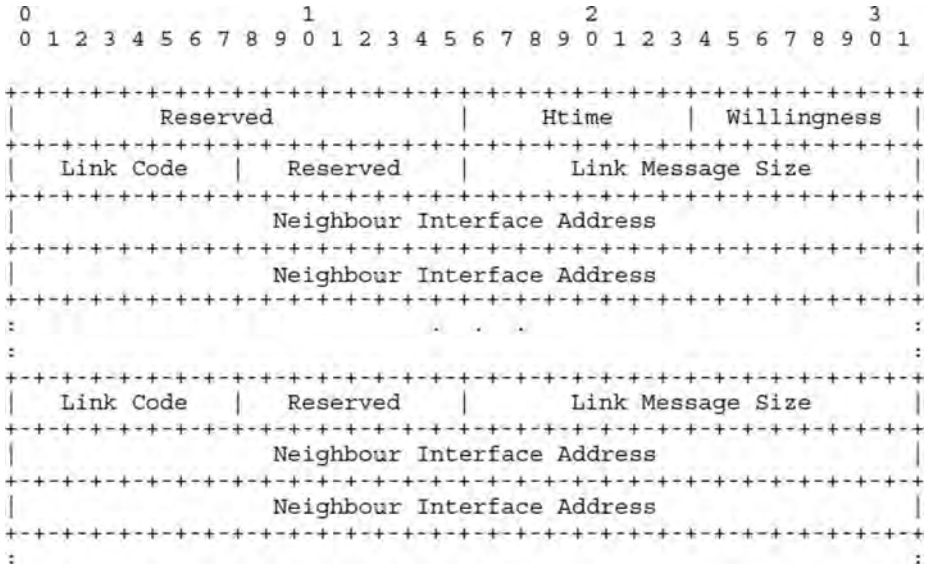


Fig. 2 *HELLO message format*

doi: 10.1049/iet-ifs.2010.0085

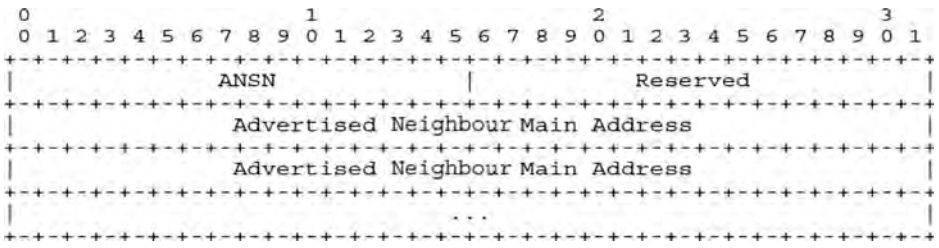


Fig. 3 *TC message format*

• *Incorrect traffic generation (ITG)*: This attack generates incorrect traffic although the messages are correct. There are two types: (i) 'Replay' attack, in which an intruder node forwards obsolete messages generated in the network and (ii) 'Relay' attack (Wormhole attack) [11], difficult to detect, in which two or more nodes that collaborate to create a tunnel between them have direct access to the network. Once the tunnel has been created (worm hole), attackers encapsulate their ingoing messages and carry out the exchange of these messages through the tunnel, preventing intermediate nodes from receiving control messages.

1.2 COD-optimised link state routing

The proposed extension, called COD-OLSR, aims to detect incorrect message generation, making spoofing more difficult. The COD-OLSR mechanism consists of two phases: coding and verification. These two phases are interrelated, so that the integrity of an encoded message in the sender is checked by the receiver, as shown in Fig. 4. Phase encoding uses binary operations (complement to one and XOR) that transform the information into 32-bit numbers. In this encoding process a series of functions are used allowing three fields to be generated which are then added to the control message header as shown in Fig. 5. These fields are: COD originator address, COD links and COD random, for a total of 12 bytes, so that the overhead introduced is negligible. Each of these fields has a specific functionality. Thus, in order to prevent identity spoofing, the message must generate COD originator address as shown in Fig. 6. This field is created by GenerateCodOA function [see (1)], using the following parameters: the main direction of node (main address), which is the network address that configures to the node, a message sequence number MSN of two bytes and a random number random of 32 bits generated by the function GenerateRandom. This last function generates numbers taking into account the current system and therefore providing greater randomness.

$$\text{GenerateCodOA} = \overline{\text{OA} \oplus \overline{\overline{\text{RND}}} \oplus \text{MSN}} \tag{1}$$

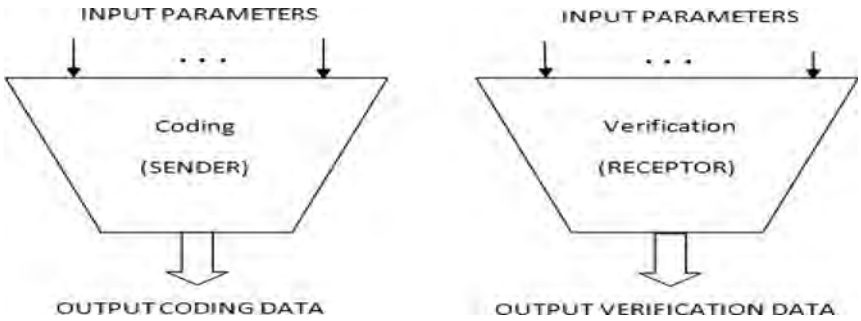


Fig. 4 *Coding and verification process*

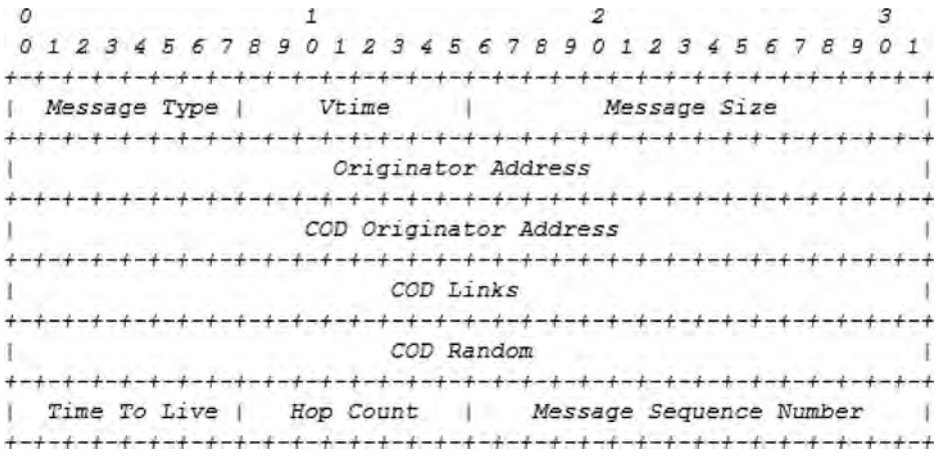


Fig. 5 *COD-OLSR message header format*

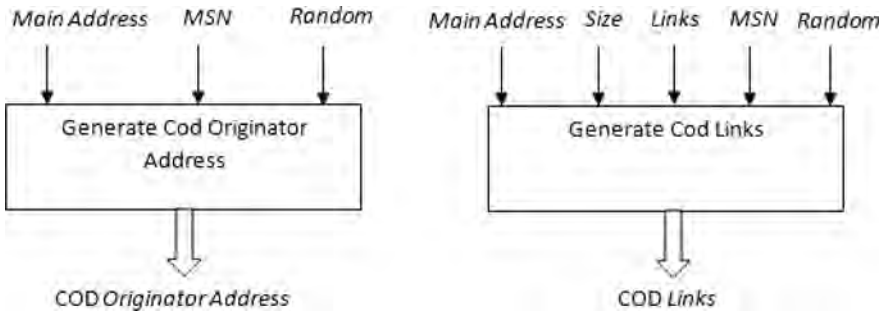


Fig. 6 *COD originator address and COD links generation*

where OA, originator address; RND, random number generated by GenerateRandom function; MSN, message sequence number.

Similarly, in order to prevent link spoofing, a COD links field is used, which is encoded using the GenerateCodLink function as shown in Fig. 6. This field is constructed by combining several parameters [see (2)]: the main direction of node (main address), the value of the message size (size, 16 bits), the same random number random mentioned above, and a list of 32-bit values associated with the content of control messages.

$$\text{GenerateCodLink} = \overline{\text{OA} \oplus \overline{\overline{\text{RND}}} \oplus \text{MSN} \oplus \text{SIZE} \oplus \overline{\overline{\text{LINK}}}} \tag{2}$$

where SIZE, packet size (bytes); LINK, 32-bit value encoded with the elements of the list.

This list of 32-bit values associated with the content of control messages can be associated with the addresses of the neighbours if it is a HELLO control message or addresses of MS in the case of TC control messages.

This list is encoded from N to 1 [see (3)], that is, several items are transformed into one through the binary operations mentioned above.

$$\text{LINK} = \begin{cases} \text{link}_{i-1} \oplus \overline{\text{link}_i} & \text{if } i \text{ is even} \\ \text{link}_{i-1} \oplus \text{link}_i & \text{if } i \text{ is odd} \end{cases} \tag{3}$$

where link_{i-1}, the previous value of the list; link_i, the current value of the list.

Finally, in order to increase protection, the COD random field is provided. This field is created with the GenerateCodRandom function [see (4)]. As shown, the parameters used are: originator address, the previous message sequence number MSN, random number random decoded and a list of values (IP address of nodes) that are the key of COD-OLSR. This list of 32-bit values represents the current topology with regard to the node that generates the control message, which also has an encoding N to 1.

$$\text{GenerateCodRandom} = \overline{\text{OA} \oplus \text{RND} \oplus \overline{\overline{\text{MSN}}} \oplus \text{LINK}} \tag{4}$$

As mentioned above, the attack detection is performed in the receiver (in the verification phase), which checks that the fields encoded in the sender have not been altered, ensuring the integrity of messages.

In the example shown in Fig. 7, we see how both the sender and the receiver nodes take the topology into account. The sender node C sends a HELLO control message to node D. One hop neighbours of C are topology = $\{A, B, D\}$. When

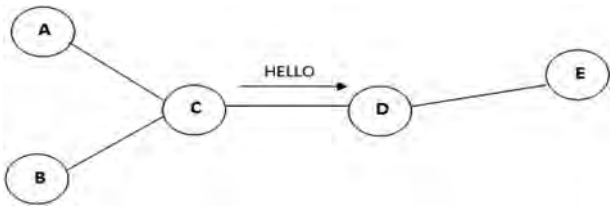


Fig. 7 *Scheme of topology coding*

the receiver D receives this message HELLO the topology of sending node C must be calculated in the verification process. Thus, it calculates the two-hop neighbours, with the intermediate node being C. To avoid problems of inconsistency (since the sending node topology may change after sending the message), we used a technique of synchronisation so that the elapsed period is very small (in the order of milliseconds) from the message coding phase in the sender to the verification in the receiver. In the verification process, the function RecoverCodRandom is used [see (5)].

$$\text{RecoverCodRandom} = \text{OA} \oplus \overline{\text{MSN}} \oplus \text{LINK} \oplus \overline{\text{COD}} \tag{5}$$

Thus, the receiver calculates which neighbours the sender has. Once the receiver has found the topology of the sending node, the reverse process is performed: first the random number random using this function is calculated, which retrieves the original random number generated by the sender, by combining the following parameters: COD random, originator address, MSN and topology. Then with this random number random and with information of message received by the receiver, it calculates COD originator address and COD links as seen in the coding phase. If the calculated values do not correspond to the fields COD originator address and COD links contained in the message, the receiver decides that an attack has taken place and it discards the message.

Fig. 8 shows a diagram for the generation of COD random. The most important feature of this diagram is that it uses the encoding of the current topology of the sending node, so that the suspected attacker, ignoring the dynamic topology of the network, although he can see the message, cannot interpret it. If the topology of the sending node does not change, the attacker can exploit this situation by monitoring the network and then performing the spoofing. As a security measure, for the coding of the three fields COD originator address, COD links and COD random the message sequence number MSN is used.

doi: 10.1049/iet-ifs.2010.0085

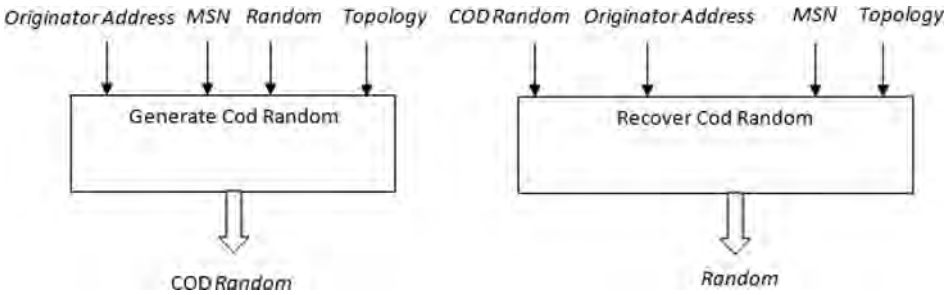


Fig. 8 *Generate and recover of random*

1.3 Simulation

In this section we analyse the behaviour of COD-OLSR in face of OLSR. We also check the efficiency of COD-OLSR with different percentages of attackers and in different situations to see how the most representative performance metrics, such as overhead, system failure tolerance, delivered data packet ratio and throughput evolve.

1.4 Performance metrics

In this section we define the most important performance metrics used in the simulation:

- *Overhead in number of packets*: Relationship between the total numbers of transmitted control packets by the nodes of the network and the number of delivered data packets to their destinations.
- *Overhead in number of bytes*: Relationship between the total numbers of transmitted control bytes and delivered data bytes.
- *System failure tolerance*: This metric takes into account the number of control messages that are discarded by COD-OLSR.
- *Delivered data packet ratio*: Relationship between number of sent packets and the number of delivered packets successfully.
- *Throughput*: Volume of work or information flowing through a system. Is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.

2 Results

In the simulation we utilised the Network Simulator NS-3 [12]. We have carried out two kinds of experiments: the first experiment compares the overhead between COD-OLSR and OLSR. The second one consists of a study of COD-OLSR against various attacks. In this second experiment, the results have been obtained according to various performance metrics. In both experiments the following assumptions were considered: (i) the intruder nodes do not carry out simultaneous attacks in HELLO and TC control messages, (ii) the attack in question is either identity spoofing or link spoofing, not both kinds of attacks in the same message. The following features are shared by both experiments: we have used an area of 1000×1000 m^2 and a simulation time of 30 s. Four data sessions have been considered (always the same source–destination pairs). The application used for data generation has been constant bit rate, in which four packets of 64 bytes of data per second are sent, that is a data transmission rate of 2048 bps. The used mobility model has been random waypoint with a speed between a minimum of 0 and a maximum of 5 m/s, with pause time of 5 s. Network devices were configured according to the standard IEEE 802.11b with a transmission range of 150 m.

2.1 Comparison of the overhead between COD-OLSR y OLSR

In this comparison a variable density of nodes between 20 and 100 nodes has been used and we compared the overhead of the two protocols without considering any type of attack in order to see if the COD-OLSR overhead is insignificant. We draw the comparison with both the number of packets and of bytes because they do not behave in the same way, since the first considers the number of messages per packet, while the second refers to the total size of the packet in bytes.

2.2 Overhead in number of packets

Fig. 9 shows the comparison of the overhead in the number of packets between COD-OLSR and OLSR. We see that in both protocols the overhead is very similar; nevertheless, in very dense networks COD-OLSR has a slight overhead compared to OLSR.

2.3 Overhead in number of bytes

In Fig. 10, we see how the overhead in the number of bytes is a bit higher in COD-OLSR than in OLSR when we increase

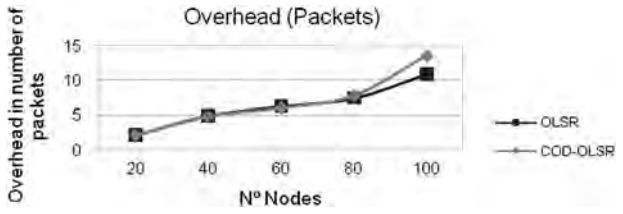


Fig. 9 *OLSR against COD-OLSR according to overhead in packets*

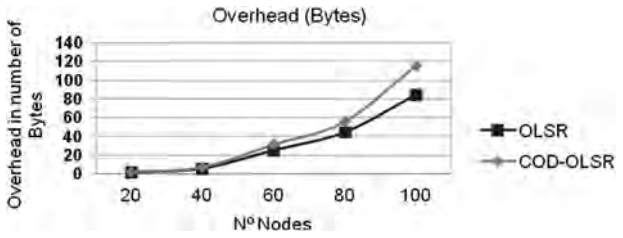


Fig. 10 *OLSR against COD-OLSR according to overhead in bytes*

the density of nodes. This is because the header control messages of COD-OLSR have three more fields (4 bytes each one), so that the protocol has to process 12 bytes more.

2.4 Study of COD-OLSR

In this section we analyse the behaviour of the proposed extension of OLSR with attacks of different characteristics, to verify that the stability of the protocol is not altered.

2.5 System failure tolerance

The system failure tolerance is the fraction of lost control messages (not processed by the algorithm). In this experiment 100 nodes for the representation of the graph are used.

In Fig. 11, we see that the system failure tolerance depends more on the number of intruders than the number of attacked messages. We appreciate in this graph that if there is a 100% message attack with 20% intruders, packet loss is slightly over 12%. We also see that, when the number of intruders decreases to 5%, the loss of messages does not exceed 2%, whatever the percentage of attacked messages may be. These results make us see that COD-OLSR has a good system failure tolerance, providing stability to OLSR.

2.6 Delivered data packet ratio

In Fig. 12, we see the delivered data packet ratio, taking into account that malicious nodes attacks to control messages are 100%. Thus, we observe the behaviour of the system in the most extreme case. We note that the ratio depends on increasing the density of nodes. In sparse networks (20 nodes) the resulting line is nearly uniform, held within the 20% ratio, regardless of the number of attackers, while in very dense networks (100 nodes), the ratio decreases from 70 to 20% when we increase the number of intruders, since the attacks provoke incorrect routing.

2.7 Throughput

The throughput behaves similar to the ratio, as shown in Fig. 13. These results make us see that both throughput and

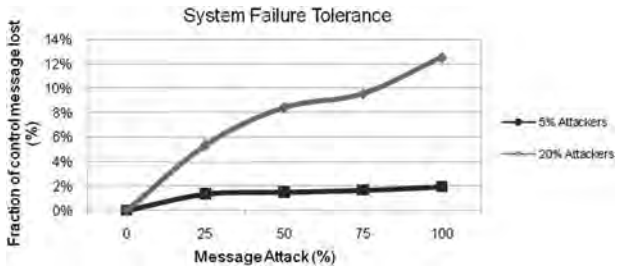


Fig. 11 *System failure tolerances (COD-OLSR)*

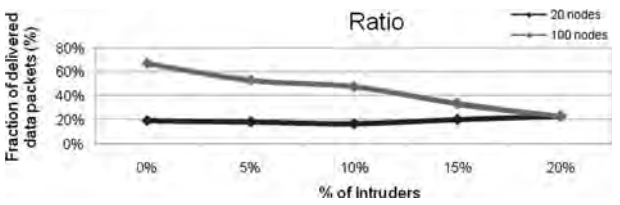


Fig. 12 *Delivered data packet ratio (COD-OLSR)*

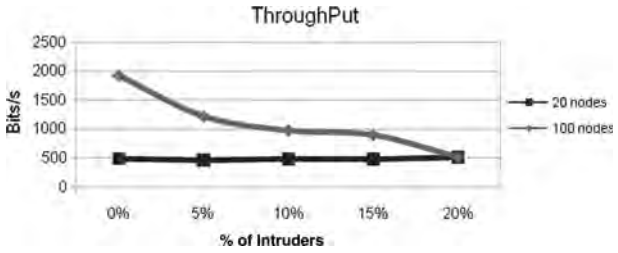


Fig. 13 *Throughput (COD-OLSR)*

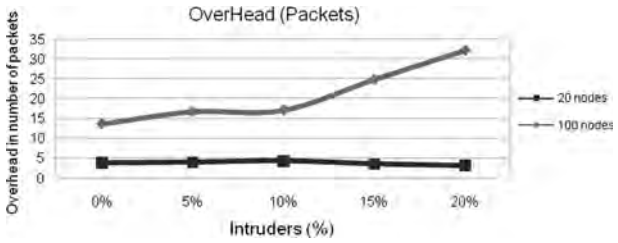


Fig. 14 *Overhead in packets (COD-OLSR)*

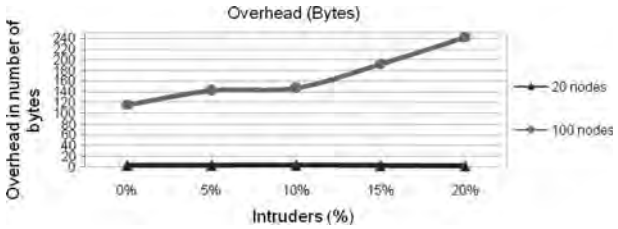


Fig. 15 *Overhead in bytes (COD-OLSR)*

delivered data packet ratio metrics are closely related, but they use different scales.

2.8 Overhead (packets)

In Fig. 14, we analyse the behaviour of overhead in number of packets according to the density of network nodes with 100% of attack messages. In sparse networks (20 nodes) it does not overcome five packages, and so it presents a uniform line whatever the percentage of attacks is. When, we increase the number of intruders in very dense networks (100 nodes), the overhead in the number of packets increases dramatically. We see that with 20% of attackers, overhead reaches a value of more than 30 packets.

2.9 Overhead (bytes)

Under the same conditions as in the previous case, that is, Fig. 14, the behaviour of overhead in the number of bytes is very similar, as shown in Fig. 15. In sparse networks, the overhead in number of bytes is practically negligible whatever the percentage of attackers. In contrast, it increases considerably in very dense networks when the number of attackers increases.

3 Conclusions

Routing protocols for MANETs do not take into account the hostile environments, so that security extensions should be added. We analyse OLSR, one of the most important routing protocols, which does not have security in its specification, presenting an extension to this protocol, called COD-OLSR,

doi: 10.1049/iet-ifs.2010.0085

which protects it from incorrect message generation, both identity spoofing and link spoofing. One of the most important characteristics of this extension is that it takes into account the current topology of the sending node, so that the receiver of the message can verify the integrity of control messages. The obtained results show that COD-OLSR has a slight overhead, but without being detrimental for the performance of OLSR protocol, ensuring integrity as seen in the performance metrics. We also see that this method is an alternative to security using cryptography, in which there is a high overhead. For further research, we are working on adapting our framework to other kinds of vulnerabilities, such as 'Replay' attacks and Wormhole attacks.

4 Acknowledgments

This work was supported by the Ministerio de Ciencia e Innovación (MICINN) through the Projects TEC2007-67129/TCM and TEC2010-18894/TCM and the Ministerio de Industria, Turismo y Comercio (MITyC) through the Project Avanza Competitividad I + D + I TSI-020100-2010-482.

5 References

1 IETF: 'Mobile ad-hoc networks (MANET) working group'. Available at http://www.ietf.org/html.charters/manet-charter.html

2 Clausen, T., Jacquet, P.: 'Optimized link state routing protocol (OLSR)'. IETF RFC3626, 2003, available at http//www.ietf.org/rfc/rfc3626.txt

3 Halfslund, A., Tonnesen, A., Rotvik, R.B., Andersson, J., Kure, O.: 'Secure extension to the OLSR protocol'. OLSR Interop and Workshop, 2004

4 Adjih, C., Clausen, T., Jacquet, P., Laouiti, A., Mühlethaler, P., Raffo, D.: 'Securing the OLSR protocol'. Proc. Med-Hoc-Net, Mahdia, Tunisia, June 2003

5 Raffo, D., Clausen, T., Adjih, C., Mühlethaler, P.: 'An advanced signature system for OLSR'. ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN-04), Washington, DC, USA, October 2004

6 Raffo, D., Adjih, C., Clausen, T., Muhlethaler, P.: 'OLSR with GPS information'. Proc. 2004 Internet Conf. (IC 2004), Tsukuba, Japan, October 2004

7 Papadimitratos, P., Haas, Z.J.: 'Secure link state routing for mobile ad hoc networks'. Proc. 2003 Symp. Applications and the Internet Workshops (SAINT'03 Workshops), DC, USA, January 2003, pp. 379–383

8 Nait-Abdesselam, F.: 'Detecting and avoiding wormhole attacks in wireless ad hoc networks', *IEEE Commun. Mag.*, 2008, **46**, (4), pp. 127–133

9 Abusalah, L., Khokhar, A., Guizani, M.: 'A survey of secure mobile Ad Hoc routing protocols', *IEEE Commun. Surv. Tutor.*, 2008, **10**, (4), pp. 78–93

10 Kannhavong, B., Nakayama, H., Kato, N., Jamalipour, A., Nemoto, Y.: 'A study of a routing attack in OLSR-based mobile ad hoc networks', *Int. J. Commun. Syst. (IJCOMSYS)*, 2007, **20**, (11), pp. 1245–1261

11 Hu, Y.C., Perrig, A., Johnson, D.B.: 'Wormhole attacks in wireless networks', *IEEE J. Sel. Areas Commun. (JSAC)*, 2006, **24**, (2), pp. 370–380

12 The ns-3 network simulator. Available at http://www.nsnam.org

Journal of Ubiquitous Systems & Pervasive Networks
Volume 3, No. 2 (2011) pp. 67-72

An Extension Proposal of AntOR for Parallel Computing

Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba*

Grupo de Análisis, Seguridad y Sistemas (GASS)
Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA)
Facultad de Informática, Despacho 431
Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n
Ciudad Universitaria, 28040 Madrid, Spain
E-mail: {delfinrc, asandoval, javiergv}@fdi.ucm.es

Abstract

Designing routing protocols for mobile ad hoc networks (MANETs) is a complex task because of its dynamic topology. A kind of routing protocols that suits the particularity of mobile ad hoc networks is so-called bio-inspired. Among these, focused on Ant Colony Optimization (ACO), which studies the behaviour of ants in their search for food, are especially relevant. One of these algorithms is AntOR, which relying on swarm intelligence, efficiently solves routing in mobile ad hoc networks. AntOR is a hybrid ACO routing protocol and can be considered as a variant of the AntHocNet protocol, which improves the performance of it in important parameters such as delivered packet ratio, the overhead in the number of packets and the overhead in the number of bytes. The protocol is stable in the carried out simulations, which is suggested in its scalability. In this article we propose an extension of AntOR that using programming multiprocessor architectures based on shared memory protocol, allows to run tasks in parallel using threads, being applicable this parallelization in the route discovery phase, route local repair process and link failure notification.

Keywords: *Mobile Ad Hoc Networks, Routing Protocol, ACO, AntOR, Extension, Parallel Computing.*

1. Introduction

A Mobile Ad hoc NETwork (MANET) [1] is a group of mobile nodes which operate themselves through wireless links. In contrast with conventional networks, a mobile ad hoc network doesn't need an infrastructure, since nodes rely on each other to operate themselves, forming what is called: multi-hop communication. Such networks have more problems and disadvantages than a conventional network because the topology of mobile networks may change quickly and without warning. As a result, errors in the transmission are unpredictable and a lack of security is common, both in nodes and links. We must add that the nodes have limited resources since; an ad hoc network is usually made by battery-powered devices with low capacity, memory, and so on.

Basically, routing protocols based in MANETs are classified into three categories: proactive, reactive and hybrid. Proactive routing protocols often need to exchange control packets among mobile nodes and continuously update their routing tables. Each node must maintain the state of the network in real time. This causes high overhead congesting of the network, which requires lots of memory. The advantage of proactive protocols is that nodes have correct and updated information.

So, when we need a path, we can find it directly in memory and establish links quickly.

These protocols are intended to reduce broadcasting frequency whilst maintaining the correct information for the routing table. Reactive routing protocols only seek a route to the destination when it is needed. The advantage of these protocols is that the routing tables located in memory are not continuously updated. On the other hand, they have the disadvantage that they can´t establish connections in real time. The aim of these protocols is to save time in the route discovery process, since the reactive protocol is designed to reduce the latency which is critical in this kind of protocols. It also aims to avoid the maintenance of routes to produce long delay.

Hybrids are derived from a mixture of these two protocols, and for this reason, they share some of their advantages. This article, which shows an extension of an innovative hybrid routing algorithm, the so-called AntOR [2], is organized as follows: in section 2, we present related work, where we expose some of the bio-inspired protocols based specifically on

* Corresponding author. Tel.: (+34) 91 394 7638
Fax: (+34) 91 394 7547. E-mail: javiergv@fdi.ucm.es

DOI: 10.5383/JUSPN.03.02.005

the ant behavior, Ant Colony Optimization (ACO), for Ad Hoc networks. In Section 3, we expose a base protocol which supports our approach, describing and analyzing it to make a comparative study. In Section 4 we explain the features of AntOR in detail. In Section 5, the extension proposal of AntOR for parallel computing is presented. Finally, the paper concludes in Section 6 with overall conclusions, observations and potential advancements for further investigations.

2. Related work

Mobile ad hoc networks have special characteristics that must be taken into account when a routing protocol is implemented. There are many solutions (RFC 3626 [3], RFC 3561 [4] RFC 4728 [5], RFC 3684 [6]...). All these protocols are valid solutions, but they usually have a specific topology and characteristics of certain scenarios as a design basis. They are not always particularly suitable if there are drastic changes in the dynamic topology of the network.

There is a group of algorithms or routing protocols called bio-inspired whose essential characteristic consists of being adaptive, which is especially noteworthy in this kind of network. The concept of Swarm intelligence [7], is specifically referred to in literature. It is based on the application of social behavior of insects and other animals to solve problems.

The Ant Colony Optimization (ACO) [8] algorithm is the starting point of these algorithms. ACO algorithms are based on the collective behavior of ants in search of food to bring back to the nest. Various tasks are performed by proposals of ACO routing, in which proactive, reactive and hybrid protocols are found.

Since proactive ACO routing protocols are included: Probabilistic Emergent Routing Algorithm (PERA) [9], has a low delivered data packet ratio in scenarios with high mobility, it has a high overhead caused by control messages being sent several times in broadcast mode.

Another protocol is Ant Routing Algorithm for Mobile Ad hoc networks (ARAMA) [10] which, according to the authors, reduces the overhead of discovery and maintenance of the routes; but, they do not discuss how they control the generation of control messages in a dynamic environment. However it has the common characteristic of achieving a low latency in the route discovery process, with the information of the routing table receiving correct updates.

We also mention reactive protocols:

Ant-Colony-Based Routing Algorithm (ARA) [11], this approach is made use of the process of flooding to update pheromone tables in all nodes. This process has greater scope in the transmission of packets than a simple broadcast, but leads to high overhead. ARA is not scalable and does not detect loops.

Ant Colony based Multi-path QoS-aware Routing (AMQR) [12], this protocol is robust and can withstand better Quality of Service (QoS), but similarly to reactive protocol, it has a high latency in the discovery of routes.

As hybrid ACO routing protocols are included:

Ant-AODV [13], in which the latency of route discovery is reduced, because the process of route discovery is reduced. Hybrid ant colony optimization routing algorithm for mobile ad hoc NETwork (HOPNET) [14], is a highly scalable protocol, with the disadvantage that, when the number of nodes is low, the continuous movement of the peripheral nodes incites to discover new routes causing more delay than other hybrid protocols.

AntHocNet [15] [16] [17] protocol, that is based on the approach made in this article. This protocol does not take into account disjoint-link/node routes, and has a high overhead in the process of exploring new routes. The disadvantage of the previously reviewed protocols is not using disjoint routes, whether link or node.

3. AntHocNet

AntHocNet [16] [17] is a hybrid ACO routing algorithm. Data from 2004 and 2005 has had numerous extensions and a great impact, being a pioneer algorithm in this field. This protocol is applied to multipath and dynamic networks, that is, creating multiple paths to transmit data from source to destination in the same data session. AntHocNet follows a structure similar to AntNet-FA (Fast Ant) [18], but it differs from AntNet-FA given that topologies of static networks are applied and convergence is slow. So, all ants have to do is choose the path. AntHocNet, meanwhile, takes into account the dynamic topology and other characteristics of ad hoc networks. When the network topology changes, then it must be restored quickly and this is achieved through a new route discovery process. If multiple resources are used to accelerate this process, the exchange of information is enhanced. This can cause the network to collapse. Therefore, we have a problem: If we do not want to overload the network, we increase the convergence time (time required for the network to become stable before a link failure) of the ACO algorithm, and if we want to reduce the convergence time, overload the network. The previous problem means that AntNet-FA algorithm does not directly apply to mobile ad hoc networks, so it needs a modification to accelerate its convergence time without overloading the network. In this way, AntHocNet emerges as a reactive, adaptive, multipath and proactive algorithm (hybrid). It is reactive because it has agents operating on demand to setup routes to destinations.

It is proactive because it has agents collecting information and these agents can discover new routes which serve as alternatives if a link fails.

It's multipath because it provides different routes to send information to the destination.

It is adaptive because it adapts to traffic conditions and networks.

In the operation of AntHocNet, the following stages or phases are distinguished:

- Routing information Setup: The source node sends reactive agents to discover the first available route to the destination.
- Data Routing: Data is sent through the nodes to the destination using the route information and can use a multi-hop technique, which involves sending data through intermediate nodes. These nodes act as routers.
- Path maintenance and exploration: Information about existing routes is proactively updated and the discovery of new ones is possible.
- Management of link failures: Management failures occur when a node is outside the scope of the network or does not receive control messages which are responsible for informing a node of its closest neighbors (who are one hop), and so on. This phase deals with such failures.

4. AntOR

AntOR is a hybrid ACO routing protocol that takes Ducatalle algorithm [19] as its starting point, and demonstrates the following qualities and abilities, which they are different from AntHocNet:

- Disjoint-Link and Disjoint-Node protocol.
- Separation between the pheromones in the diffusion process.
- Use distance metric in path exploration.

4.1. Disjoint-Link/Node protocol

In such protocols there are two kinds of routes: Disjoint-Node and Disjoint-Link. The first corresponds to routes in which nodes aren't shared and the latter refers to routes in which links aren't shared. Every Disjoint-Node is also Disjoint-Link, but not vice versa.
Both types of disjoint routes have the following advantages:

1. A failure in one node only affects a path, not the entire network.
2. Load balancing is better because there aren't repeated routes on the disjoint property.

However, the use of such routes does also have its disadvantages, for example:

1. More resources are needed because they do not share links and nodes.
2. These routes are more difficult to detect, because we limit the nodes that can be explored.

Our approach does not use Disjoint-Partial routes as in [20]. Disjoint-Link and Node are different from the Disjoint-Partial because they may have both nodes and links in common. Disjoint-Partial routes are less restrictive, allowing computing of more routes, providing better tolerance to link failures as a result of quick and efficient recovery from broken routes. Also, in general, they are easier to detect. But they have the disadvantage of using some intermediate node of the main route (hybrid approach between Disjoint-Link and Disjoint-Node), so whether an intermediate node fails a new route has to be discovered. Therefore, the routes which are only Disjoint-Link or Disjoint-Node prevent better link failures because they have alternatives so that nodes and links aren't repeated. This is a reason not to use Disjoint-Partial routes.
Next we will show how to calculate Disjoint-Node and Disjoint-Link with regard to AntOR algorithm.

4.1.1. Disjoint-Node Routes

As mentioned above, a Disjoint-Node route is one in which nodes are not shared in the same data session. Fig. 1 shows an example of Disjoint-Node routes.

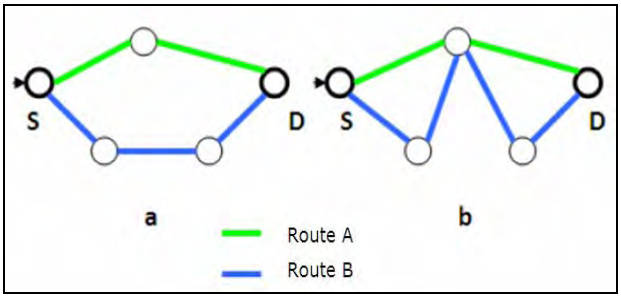


Fig. 1. Node-Disjoint Routes (a) against no Node-Disjoint Routes (b)

The procedure for calculating Disjoint-Node route is different from how we calculate Disjoint-Link. In the case of Disjoint-Node, the intermediate nodes which have been visited are marked so we do not repeat them. This table shows information regarding the previously visited nodes is stored "locally". This aims to perform a proactive process for the exploration of new routes.
In Fig. 2 a flow chart shows the functionality of this method.

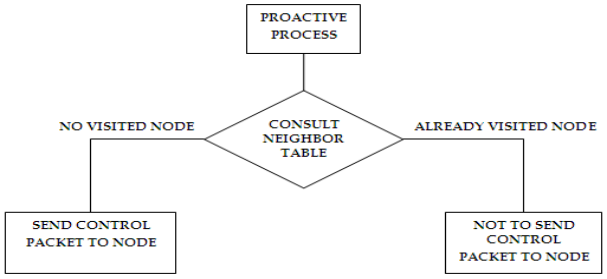


Fig. 2. Operation of Node-Disjoint Route

4.1.2. Disjoint-Link Routes

As previously mentioned, a disjoint-link route is one in which no links are shared on a same data session. Figure 3 shows an example of disjoint-link route, which recognizes the difference between Disjoint-Link route and Non Disjoint-Link routes. The basic idea for finding and representing Disjoint-Link routes is to mark each disjoint link with a label indicating what the source of the data session is.

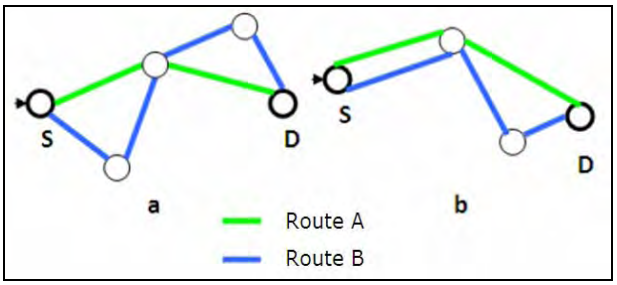


Fig. 3. Link-Disjoint Routes (a) against no Link-Disjoint Routes (b)

In Fig. 4 we present a diagram showing how this process functions. We can see that the mechanism is very simple.
When two or more data sessions sharing links in its information retransmission, the procedure to calculate Disjoint-Link routes is similar: each link which is visited is marked according to the session that it belongs to. As a result, there will be no conflict or problems since the sessions do not rely on each other.

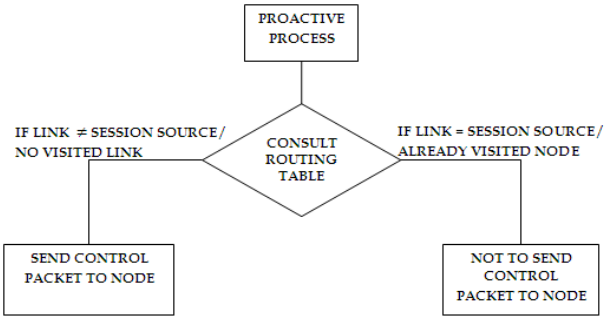


Fig. 4. Operation of Link-Disjoint Route

4.2. Separation between the pheromones in the diffusion process

In Ducatelle's approach [18], the same route can have both regular pheromone values and pheromone virtual values simultaneously.
Regular pheromone values are used to determine the routes through which data travels. On the other hand, virtual pheromone values indicate the routes that can possibly be "good" to relay the data.
In AntOR, a route cannot have both a regular pheromone value and a virtual pheromone simultaneously; this technique improves the efficiency of the algorithm. To carry out this separation, the Equation (1) of Ducatelle's route exploration has changed [18].

$$P_{in}^{d} = \frac{\max\left(\tau_{in}^{d}, k_{in}^{d}\right)^{\beta_2}}{\sum_{j \in N_i^d} \max\left(\tau_{ij}^{d}, k_{ij}^{d}\right)^{\beta_2}} \quad (1)$$

Where:

P_{in}^{d} : Is the likelihood that node i chooses the next hop n with destination node d.

max (a,b): This function returns the maximum of two values a and b.

τ_{in}^{d} : It corresponds to the regular pheromone value.

k_{in}^{d} : It corresponds to the virtual pheromone value.

β_2 : It's a protocol configuration parameter. It is used in the proactive process for new routes exploration. When this parameter has a low value, all alternative routes to a destination can be chosen with equal probability.

N_i^d : This is a one hop neighbour of node i.

The result of this modification of Equation (1) is shown in the following one.

$$P_{in}^{d} = \frac{(\psi_{in}^{d})^{\beta_2}}{\sum_{j \in N_i^d} (\psi_{ij}^{d})^{\beta_2}} \qquad \psi \in \begin{cases} k & \text{virtual} \\ \tau & \text{regular} \end{cases} \quad (2)$$

Where:

ψ : It corresponds to a regular or virtual pheromone value, but never both simultaneously.
Then, an example to compare the difference of Equations (1) and (2) is shown. There are four nodes (A, B, C and D) in the scenario. As node A wants to send data to the destination node D, node A needs to choose the most appropriate route.

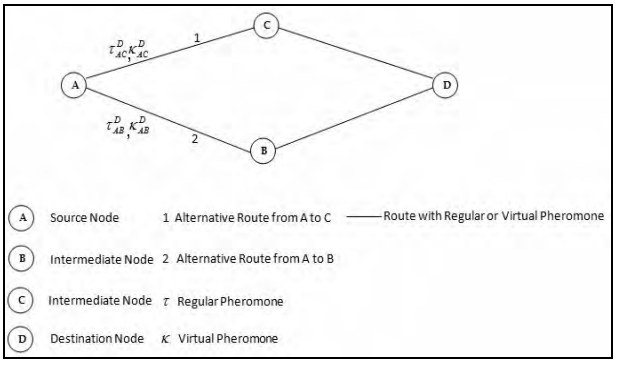


Fig. 5. Illustrative Example of Equation (1)

Figure 5 shows the scenario from the point of view of Ducatelle equation or Equation (1), where the two routes from A have regular and virtual pheromone. Therefore, to calculate the probability of choosing the alternative route 1, the following equation is used:

$$P_{AC}^{D} = \frac{\max(\tau_{AC}^{D}, k_{AC}^{D})^{\beta_2}}{\max(\tau_{AB}^{D}, k_{AB}^{D})^{\beta_2} + \max(\tau_{AC}^{D}, k_{AC}^{D})^{\beta_2}} \quad (3)$$

On the other hand, Figure 6 shows the same scenario but from the perspective of AntOR. In this case the alternative route 1 has only regular pheromone and the route 2 has only virtual pheromone. Thus, the probability of choosing the alternative route 1 is given by next equation

$$P_{AC}^{D} = \frac{(\tau_{AC}^{D})^{\beta_2}}{(\tau_{AC}^{D})^{\beta_2} + (k_{AB}^{D})^{\beta_2}} \quad (4)$$

These two Equations (3) and (4) use the pheromone in a different way to calculate the alternative routes. In the case of Equation (3) a route can have regular and virtual pheromone simultaneously. In the case of Equation (4) the same route has regular or virtual pheromone, but not both at the same time.

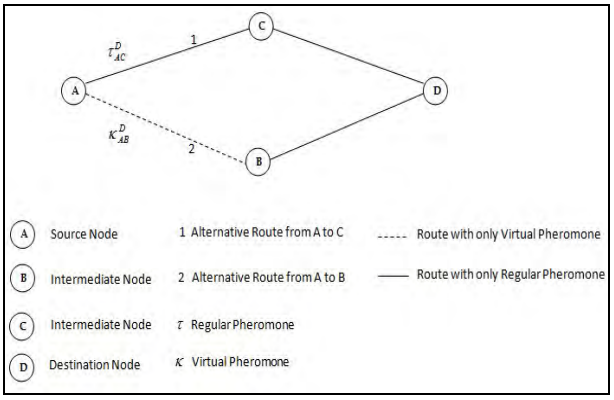


Fig. 6. Illustrative Example of Equation (2)

4.2. Use distance metric in path exploration

Our approach takes into account the number of hops for the routes which have been found to be the best. There is a hop limit on the nodes. This hop limit is established according to previously calculated routes that have a smaller distance in hop number.
Fig. 7 shows an example of how this mechanism which we have mentioned in our approach works.

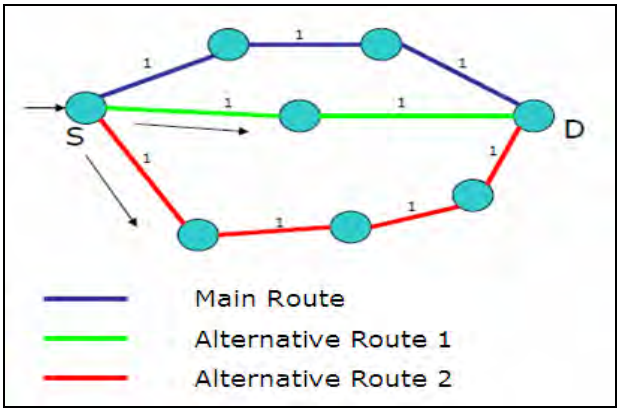


Fig. 7. Selection of Distance Metric

Once you have established the main route, sending proactive control packets to explore new routes can be chosen from two alternatives: a) Alternative Route 1 and b) Alternate Route 2. In this proactive process, Route 1 will be chosen because it has a hop count less than the main route because the main route consists of three hops. Thus, the control messages that they choose the route 2, they will never reach their destination because they are scheduled at most to visit three nodes.

5. Extension of AntOR for Parallel Computing

To understand how the extension proposal of AntOR works, it is necessary to employ three concepts:

1) Process: Program running. The processes are managed by the Operating System.
2) Thread: The basic unit of execution. Any program that executes at least has a thread.
3) POSIX Thread: Standard based in thread API for C / C++.

We use POSIX Thread because it allows a new concurrent process flow to expand. This is the most efficient multi-core systems, where the flow of processes can be scheduled to run on another processor, thus gaining speed through parallel or distributed processing. Programming with threads carries less overhead than expanding a new process, because the system does not initialize a new environment and virtual memory space for that process. Parallel programming technologies, such as MPI and PVM are used in a distributed computing environment, while the threads are limited to a single computer system. All threads within a process share the same address space. For the implementation of this routing algorithm to be faster, we use the POSIX Thread library. Then we specify a large-grained parallelization of AntOR-DNR routing algorithm (node-disjoint version). This parallel technique launching a thread for each neighbour that is in the neighbour table of the node that starts one of the following phases:

5.1. Routing Information Setup

Fig. 8 shows a flow chart representing the parallelism in the route discovery process.

When a data session is active, the source node is ready to send data to the destination node, and the route discovery process is activated. This process is parallelized using threads, so that it launches an ant (agent) reactive through an independent thread to the one-hop neighbours, with the number of utilized threads being proportional to the neighbour number of node initiating the route discovery. When an intermediate node receives this ant repeats the process. But, whether it consists in a destination node, this node sends its corresponding *Reactive Backward Ant* (RBA).

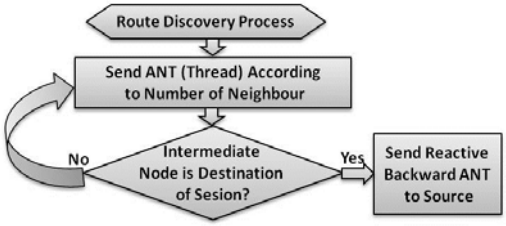


Fig. 8. Scheme to parallelize Route Discovery process

5.2. Local Route Repair

The operation is similar to route discovery, unless it is done locally, as shown in Fig. 9.

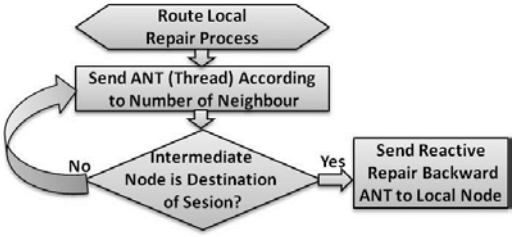


Fig.9. Scheme to parallelize Route Local Repair process

5.3. Link Failure Notification

The process of link failure notification is to update the routing table to the link failures. It is a very important stage and this action must be taken promptly. The nodes send ants through independent threads until an intermediate node has any alternative route to the destination after updating the routing table. The link failure notification process in Fig. 10 is shown.

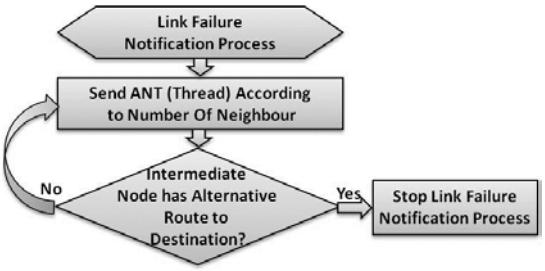


Fig.10. Scheme to parallelize Link Failure Notification process

6. Conclusions

We've presented an extension for parallel computing of a routing protocol for mobile ad hoc networks called AntOR that is classified as a hybrid ACO routing protocol (based on the algorithm of ant colony) and that can be considered as a variant of the AntHocNet protocol, which improves the performance of it in important parameters such as delivered packet ratio, the overhead in the number of packets and the overhead in the number of bytes.

The used parallel technique is a large-grained approach, in which a multicore machine in a shared memory system has been used. The essence of this parallel approach is to replace the *broadcast* messages by messages that are sent specifically to one-hop neighbours using threads.

For the future there are several lines of work, such as the comparison of AntOR with our extension and with the parallelism of link-disjoint routes.

Acknowledgments

This work This work was supported by the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) through the Project Avanza Competitividad I+D+I TSI-020100-2010-482, the Ministerio de Ciencia e Innovación (MICINN, Spain) through the Projects TEC2010-18894/TCM and TEC2010-67129/TCM and the Agencia Española de Cooperación Internacional para el Desarrollo (AECID) del Ministerio de Asuntos Exteriores y Cooperación (MAEC) through the Project MAEC-AECID C/033548/10.

References

[1] IETF. Mobile ad-hoc networks (MANET) working group. http://www.ietf.org/html.charters/manet-charter.html

[2] García LJ, Rupérez D, Sandoval AL. Bioinspired routing protocol for mobile ad hoc networks. IET Communication 2010, 4 (18): 2187-2195. http. dx.doi.org/10.1049/iet-com.2009.0826

[3] Clausen T, Jacquet P. Optimized link state routing protocol (OLSR)'. IETF RFC 3626, October 2003. http://www.ietf.org/rfc/rfc3626.txt

[4] Perkins CE, Belding-Royer EM, Das S. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC3561, July 2003. http://tools.ietf.org/html/rfc3561

[5] Jonson D, Hu, Maltz D. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. RFC 4728, February 2007

[6] Ogier R, Templin F, Lewis M. Topology Dissemination Based on Reverse Path Forwarding (TBRPF), IETF RFC 3684, February 2004. http://www.ietf.org/rfc/rfc3684.txt

[7] Kennedy J: Swarm Intelligence. Morgan Kaufmann Publishers, 2001.

[8] Dorigo M, Stützle T: Ant Colony Optimization. The MIT Press, 2004. http. dx.doi.org/10.1007/b99492

[9] Baras JS, Mehta H. A Probabilistic Emergent Routing Algorithm for Mobile Ad Hoc Networks, WiOpt 03: Modeling and optimization in Mobile Ad hoc wireless networks, Mar. 2003.

[10] Hossein O, Saadawi T. Ant routing algorithm for mobile ad hoc networks (ARAMA), Proceedings of the 22nd IEEE International Performance, Computing, and Communications Conference, Phoenix, Arizona, USA, April 2003, pp. 281-290.

[11] Günes M, Sorges U, Bouazizi I. ARA - The ant-colony based routing algorithm for MANETs. In Proceedings of the ICPP International Workshop on Ad Hoc Networks (IWAHN), 2002.

[12] Liu L, Feng G. A Novel Ant Colony Based QoSAware Routing Algorithm for MANETs. ICNC 2005, LNCS 3612, SpringerVerlag Berlin Heidelberg, 2005, pp. 457 – 466.

[13] Marwaha S., Tham C. K., Srinavasan D.: 'Mobile Agents based Routing Protocol for Mobile Ad Hoc Networks', IEEE Global Telecommunications Conference (GLOBECOM'02), Taipei, Taiwan. 2002.

[14] Wang J, Osagie E, Thulasiraman P, Thulasiram RK. HOPNET: A hybrid ant colony optimization routing algorithm for mobile ad hoc network, Ad Hoc Netw. Elsevier Science Publishers, 2009, 7 (4), pp. 690-705. http.dx.doi.org/10.1016/j.adhoc.2008.06.001

[15] Di Caro G. Ant Colony Optimization and its application to adaptive routing in telecommunication networks. PhD thesis in Applied Sciences, Polytechnic School, Université Libre de Bruxelles, Brussels, Belgium, 2004.

[16] Di Caro GA, Ducatelle F, Gambardella LM, AntHocNet: An Adaptive Nature-Inspired Algorithm for Routing in Mobile Ad Hoc Networks. European Transactions on Telecommunications, October 2005. 16, Issue 5.

[17] Di Caro G, Ducatelle F, Gambardella LM. AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks. Proceedings of PPSN VIII - Eight International Conference on Parallel Problem Solving from Nature, Birmingham, UK, September 18-22, 2004, Springer-Verlag, Lecture Notes in Computer Science, Vol. 3242.

[18] Di Caro G, Dorigo M. Two at colony algorithms for best-effort routing in datagram networks. In Proceedings of the Tenth IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS'98), IASTED/ACTA Press, 1998, pp. 541–546

[19] Ducatelle F. Adaptive Routing in Ad Hoc Wireless Multi-hop Networks, PhD thesis, Università della Svizzera Italiana, Istituto Dalle Molle di Studi sull´Intelligenza Artificiale, 2007

[20] Zafar H., Harle D., Andonovic I., Khawaja Y.: Performance evaluation of shortest multipath source routing scheme, IET Commun., 2009, 3, Issue 5, pp. 700 – 713. http.dx.doi.org/10.1049/iet-com.2008.0328

A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks

Delfín Rupérez Cañas[1], Ana Lucila Sandoval Orozco[1], Luis Javier García Villalba[1], and Tai-hoon Kim[2,3]

[1] Group of Analysis, Security and Systems (GASS)
Department of Software Engineering and Artificial Intelligence
School of Computer Science, Office 431
Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n
Ciudad Universitaria, 28040 Madrid, Spain
{delfinrc,asandoval,javiergv}@fdi.ucm.es

[2] Department of Multimedia Engineering
Hannam University
133 Ojeong-dong, Daedeok-gu
Daejeon, Korea
taihoonn@hannam.ac.kr

[3] Department of Information Technologies
Global Vision School Australia (GVSA)
20 Virgina Court, Sandy Bay
Tasmania, Australia
taihoonn@gvsa.asia

Abstract. Routing in Mobile Ad Hoc Networks (MANETs) is complex problem because of the dynamic topology, limited process and storing capability, bandwidth constraints, and lack of a centralized system. The design of efficient routing protocols is a fundamental problem in MANETs. Researchers have proposed a number of routing protocols in literature. This survey treats about a bio-inspired routing protocol called AntOR for these networks. Thus, one of its key aspects is disjoint route property. The simulation results show that the disjoint-link property has a better performance than disjoint-node according to metrics such as average End-to-End Delay and Jitter.

Keywords: Ant Colony Optimization, AntOR, Mobile Ad Hoc Network, Routing Protocol.

1 Introduction

Mobile Ad Hoc Networks (MANETs) [1] are a collection of mobile nodes which have no fixed infrastructure. The nodes communicate through wireless network and there is no central control for the nodes in the network. Routing is the task of forwarding data packets from a source to a destination. Fundamentally, routing protocols based in

T.-h. Kim et al. (Eds.): MulGraB 2011, Part II, CCIS 263, pp. 300–304, 2011.

MANETs are composed into three categories: proactive, reactive and hybrid. Proactive routing protocols often need to exchange control packets among mobile nodes and continuously update their routing tables. Each node must maintain the state of the network in real time. Reactive routing protocols only seek a route to the destination when it is needed. Hybrid protocols are derived from a mixture of these two protocols. Another brand of classification is derived from behavior of the some special animals or insects. Within these bioinspired protocols are based on ants. Ant Colony Optimization (ACO) [2] is a set of applicable algorithms which focuses on Intelligence Swarm [3] can resolve complex problem in an efficient manner.

This article proposes AntOR [4], an innovative routing algorithm belonging to such bioinspired protocols and in where we realize a comparative of its two versions: disjoint-link and disjoint-node routes.

The rest of the paper is organized as follows. In Section 2, we present related work. In Section 3, we expose main characteristics of AntOR. The most relevant simulation results are shown in Section 4. The conclusions are presented in Section 5.

2 Related Work

In recent years a large number of routing algorithm MANET has been proposed. These algorithms all deal with the dynamic aspects of MANETs in their own way, using reactive or proactive behavior, or a combination of both.

In the MANET, the main classification is between proactive, reactive, and hybrid algorithms. But other classifications exist too. The bioinspired routing protocols have a really important and relevant. These protocols might belong to proactive like Probabilistic Emergent Routing Algoritm (PERA) [5], to reactive like improved Ant Colony Optimization algoritm for mobile ad hoc Networks (PACONET) [6]. Finally, as both advantages of theses ones approach we have AntHocNet [7], protocol which our proposal is based on.

3 Review of AntOR

AntOR [4] it is a hybrid ACO routing algorithm, based on AntHocNet that takes Ducatelle algorithm [7] as its starting point, and has the following differences a) Disjoint-link and disjoint-node protocol, b) Separation between the pheromones in the diffusion process, and c) Use distance metric in path exploration.

In this protocol there are two kinds of routes: disjoint-node and disjoint-link. The first ones correspond to routes that do not share nodes, and the latter are routes which do not share links. It is satisfied the property that all disjoint node are also disjoint link, but not vice versa. Both types of disjoint routes present the following advantages.

a) A node failure affects only one route.
b) It has better load balancing because of disjoint property.

However, the use of this kind of routes presents the disadvantage that more resources are needed because of not to share links and nodes.

4 Simulation Results

We have used the Network simulator 3 (NS) [8] to evaluate the performance of following two versions: AntOR-DNR (disjoint-node route version) and AntOR-DLR (disjoint-link route version). We compare these approaches varying pause time from 0 to 120 s. We have considered 100 moving within an area of 1000m x 1000 m using *Random Waypoint Model* (RWP). The maximum speed of nodes we have considered in simulation is 10m/s. In the simulation we use the traffic generator *Constant Bit Rate* (CBR), with bit rate of 2048 bit/s (4 packets of 64 bytes per second) and total simulation time is set to 120 s. We show the general simulation parameters in table 1.

Table 1. Simulation parameters according to routing protocol

Number of node: 100
Dimensions of area: 1000 x 1000
Transmission de range (open area): 300m
Physical layer: configured for IEEE 802.11b
Send data ratio: constant WiFi-6 mps
Simulation time: 120s
Number of trials: 5

The first metrics which we have has in consideration is the Average End-to-End Delay. It consists in measure of accumulative effectiveness of experienced delays by packets going from source to destination.

In Fig. 1 we can see that AntOR-DLR has a better performance than AntOR-DNR. The main reason is because of the disjoint-link route version has more alternative route than disjoint-node version and whether one route fails, the algorithm will use another alternative one.

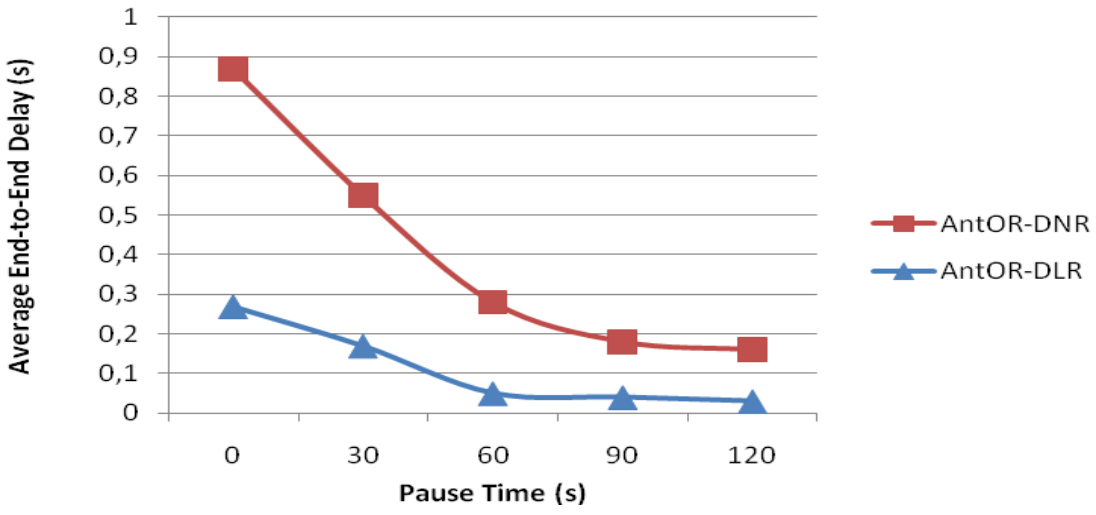


Fig. 1. Comparative AntOR-DNR against AntOR-DLR according to Average End-to-End Delay

In the Fig. 2 we can appreciate the same behavior than the Fig. 1. We see how the general performance of AntOR-DLR is better than AntOR-DNR according to Jitter. This metric takes into consideration the delay between consecutive delivered packets. We can see clearly that disjoint-link route version has a better performance to link/node failures because if a route that takes into account only the nodes and not the links, it could fails quicker and unsafely.

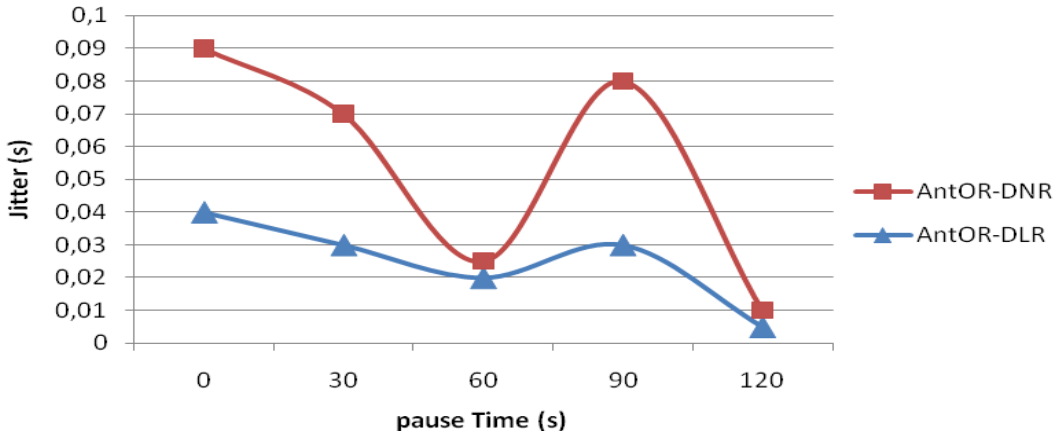


Fig. 2. Comparative AntOR-DNR against AntOR-DLR according to Jitter metric

5 Conclusions

We have presented a routing protocol for MANETs called AntOR that is classified as a bioinspired. The protocol is stable in the carried out simulations, which suggested its scalability. We can also observe the obtained results, which make the comparison in the two kinds of routes: disjoint-link and disjoint-node, where disjoint-link version has a better behavior than disjoint-node according average End-to-End delay and Jitter.

For the future, we can use these kinds of routes together from AntOR and compare results.

Acknowledgments. This work was supported by the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) through the Project Avanza Competitividad I+D+I TSI-020100-2010-482 and the Ministerio de Ciencia e Innovación (MICINN, Spain) through the Project TEC2010-18894/TCM. This work was also supported by the Security Engineering Research Center, granted by the Ministry of Knowledge Economy (MKE, Korea).

References

1. Abolhasan, M., Wysocki, T., Dutkiewicz, E.: A review of routing protocols for mobile ad hoc networks. Ad Hoc Networks 2(1), 1–22 (2004)
2. Dorigo, M., Stützle, T.: Ant Colony Optimization. The MIT Press (2004)
3. Kennedy, J.: Swarm Intelligence. Morgan Kaufmann Publishers (2001)

4. García, L.J., Rupérez, D., Sandoval, A.L.: Bioinspired routing protocol for mobile ad hoc networks. IET Communication 4(18), 2187–2195 (2010)
5. Baras, J.S., Mehta, H.: A Probabilistic Emergent Routing Algorithm for Mobile Ad Hoc Networks, Modeling and optimization in Mobile Ad hoc wireless networks (2003)
6. Osagie, E., Thulasiraman, P., Thulasiram, R.K.: PACONET: imProved Ant Colony Optimization routing algorithm formobile ad hoc NETworks. In: 22nd International Conference on Advanced Information Networking and Applications, pp. 204–211 (2008)
7. Ducatelle, F.: Adaptive routing in ad hoc wireless multi-hop networks. PhD thesis, Universitá della Svizzera Italiana, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (2007)
8. The NS-3 network simulator (2011), `http://www.nsnam.org`

Comparing AntOR-Disjoint Node Routing Protocol with Its Parallel Extension

Delfín Rupérez Cañas[1], Ana Lucila Sandoval Orozco[1], Luis Javier García Villalba[1], and Tai-hoon Kim[2,3]

[1] Group of Analysis, Security and Systems (GASS)
Department of Software Engineering and Artificial Intelligence
School of Computer Science, Office 431
Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n
Ciudad Universitaria, 28040 Madrid, Spain
{delfinrc,asandoval,javiergv}@fdi.ucm.es
[2] Department of Multimedia Engineering
Hannam University
133 Ojeong-dong, Daedeok-gu
Daejeon, Korea
taihoonn@hannam.ac.kr
[3] Department of Information Technologies
Global Vision School Australia (GVSA)
20 Virgina Court, Sandy Bay
Tasmania, Australia
taihoonn@gvsa.as

Abstract. In this paper we analysis a parallel approach of Ant-OR. This protocol is itself robust and susceptible to frequent topology changes, but with this approach called P-AntOR, which uses programming multiprocessor architectures based on shared memory protocol, we improve the behavior of AntOR, where the parallelization is applicable in the route discovery phase, route local repair process and link failure notification. The simulation results show that P-AntOR performs better than AntOR, considering metrics such as Throughput and Overhead in number of packets.

Keywords: AntOR, Mobile Ad Hoc Networks, Overhead, P-AntOR, Routing Protocol, Throughput.

1 Introduction

Mobile ad hoc networks (MANETs) [1] have had a large popularity because of their high flexibility in providing users with network access. MANET is a collection of mobile nodes that can establish quickly communication of civilian and military applications. As the size of MANET grows, the performance tends to decrease. One of the critical issues in ad-hoc networking is the lack of bandwidth and computation capability. So how to reduce the traffic overload and the pressure of computation is a very

T.-h. Kim et al. (Eds.): MulGraB 2011, Part II, CCIS 263, pp. 305–309, 2011.

important design in MANET. Many routing protocols have been proposed for efficient multi-hop routing. One of these brands consists in bioinspired protocols, existing a large variety. We focus on behavior of ants and we study the algorithms based on *Ant Colony Optimization* (ACO) [2] because of its relevance in this area. This group of algorithms or routing protocols is especially noteworthy in this kind of network due to the concept of Swarm intelligence [3]. It is based on the application of social behavior of insects and other animals to solve problems.

The routing is the handled issue in this article, so that we reference a bioinspired routing protocol in the literature called AntOR [4]. Since this starting point, we analysis an approach parallel of such protocol.

The rest of this paper is organized as follows. In Section 2 we discuss some related work. Then, we briefly present AntOR routing protocol in Section 3. This is followed by a description and results of our parallel approach in Section 4. Finally, the paper is concluded and possible future extensions in Section 5.

2 Related Work

Although ACO is itself a technique parallelizable, in this section we present most relevant parallelization techniques for ACO doing it more efficient.

One of the first one was introduced by [5], where this method can resolve difficult combinatorial optimization problems. Then, Stützle [6] applies a master/slave approximation to parallel the different searching methods from ACO solutions.

Finally, in [7] a parallelization hybrid system is shown. This method consists of evaluating the communication performance Message Passing Interface (MPI) multithreading. In this approach MPI across nodes and multithreading within a node is employed.

3 AntOR: Bioinspired Routing Protocol for MANETs

AntOR [4] has the following characteristics: a) is a protocol with the property of link/node disjoint, which provides a better distribution of packet traffic, b) has the property that separates the pheromone values in the diffusion process. Thus, a same route cannot have both a regular pheromone value and a virtual pheromone simultaneously, and c) uses the metric distance in the path exploration. This technique significantly reduces the protocol overhead.

4 Analysis of Parallel Approach of AntOR

To understand how the extension of AntOR works, it necessary to define three concepts:

a) Process: Program running. The processes are managed by the Operating System.
b) Thread: The basic unit of execution. Any program that executes at least has a thread.
c) POSIX Thread: Standard based in thread API for C / C++.

We specify a large-grained parallelization of AntOR-DNR protocol (node-disjoint version). This parallel technique launching a thread for each neighbor that is in the neighbor table of the node that starts one of the following phases: a) Routing Information Setup, b) Local route repair and c) Link Failure Notification.

These phases work as follows: The node, which initiates the process, looks for the possible neighbors in its update table and then this node sends an ant (agent) to each one through a thread. The essence of this parallelization is to substitute the *broadcast* messages by sent independent message using thread.

4.1 Simulation Results

We have executed several tests with network simulator NS-3 [8]. For the simulation we use 100 nodes configured according to IEEE 802.11b in an area of 1200 m ×1200 m, with a random distribution of the nodes. In the simulation we use the traffic generator *Constant Bit Rate* (CBR), with bit rate of 2048 bit/s and total simulation time is set to 120 s. In our experiment, that uses the Mobility Model *Random Wait Point* (RWP), we vary the node speed from 0 to 10 m/s with a pause time of 30 s.

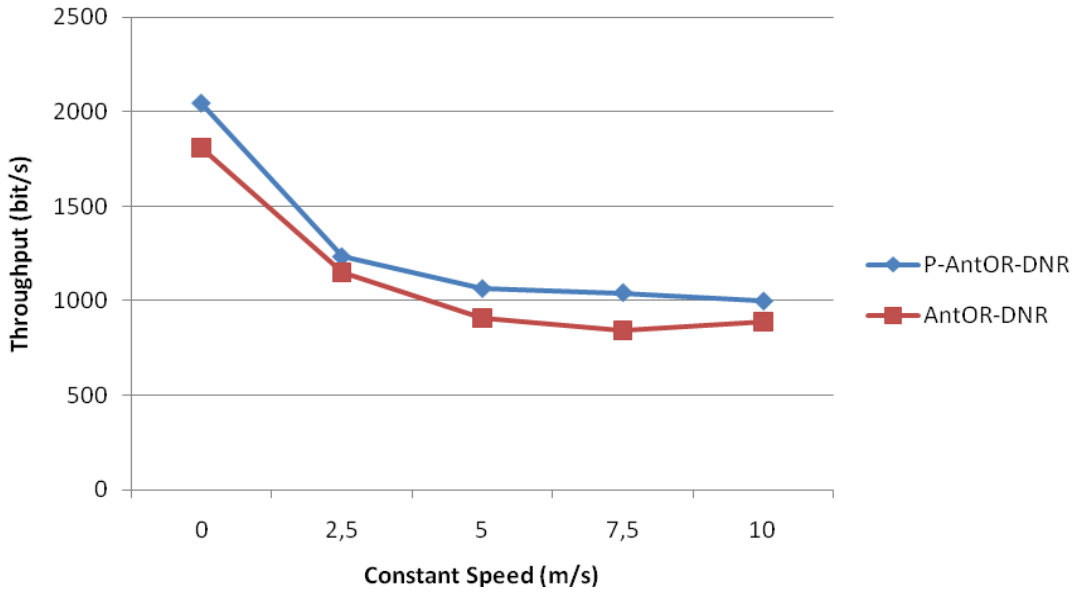


Fig. 1. Comparative according to Constant Speed against Throughput

In Fig. 1 we can see that Throughput in P-AntOR is better than its predecessor at all times, regardless of the speed of the nodes. We understand that throughput is the volume of work or information flowing through a system. It constitutes a relevant concept because of its relationship with delivered data packet ratio.

We have checked how the throughput improves with the decrease of sent packet number and we can also detect the influence of node speed.

On the other hand we analysis the overhead in number of packets in Fig. 2. It consists in relationship between the total numbers of transmitted control packets by the nodes of network and the number of delivered data packets to their destinations.

We appreciate how overhead is better in this parallel approach than Ant-OR and we show how the gap of overhead between P-AntOR and AntOR is significantly wide from 10 m/s. This improvement is due to limiting the number of sent messages, the overhead decreases.

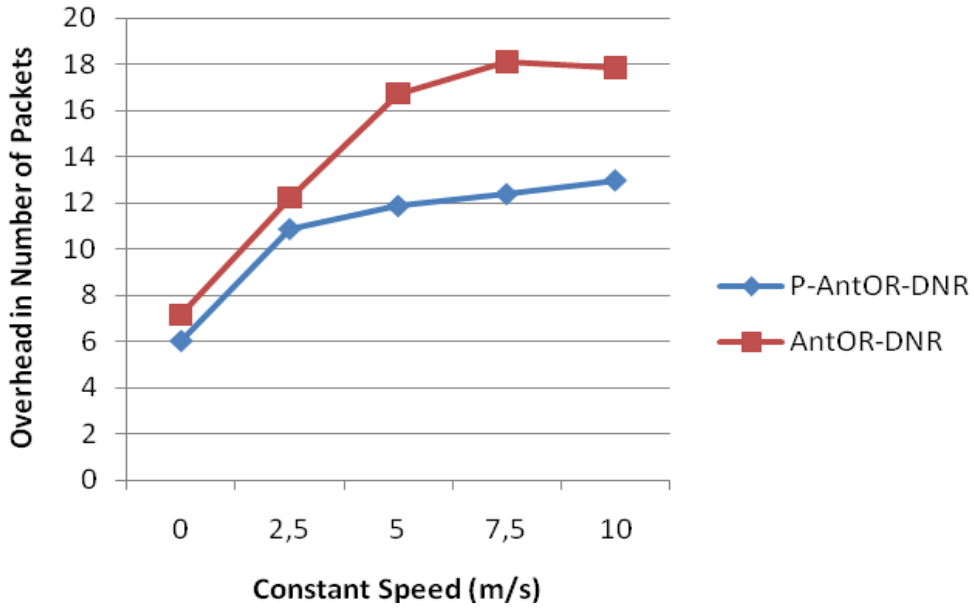


Fig. 2. Comparative according to Constant Speed against Overhead in Number of Packets

5 Conclusions

We have presented a parallel approach of AntOR, called P-AntOR. The used parallel technique is a large-grained approach. The essence of this parallel approach is to send specifically packets to one-hop neighbors using threads. According to the simulation results (Throughput and Overhead in number of packets) shows that P-AntOR improves to AntOR in their disjoint-node route version.

For the future, P-AntOR can be improved using multi-interfaces where each one of these may be handled with a multicore system. The main idea is that whether we have more “output” interface, we will send quickly more ants (agent) from each one according to parallel approach implementation.

Acknowledgments. This work was supported by the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) through the Project Avanza Competitividad I+D+I TSI-020100-2010-482 and the Ministerio de Ciencia e Innovación (MICINN, Spain) through the Project TEC2010-18894/TCM. This work was also supported by the Security Engineering Research Center, granted by the Ministry of Knowledge Economy (MKE, Korea).

References

1. Royer, E., Toh, C.: A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks. IEEE Personal Communications 6(2), 46–55 (1999)
2. Dorigo, M., Stützle, T.: Ant Colony Optimization. The MIT Press (2004)
3. Kennedy, J.: Swarm Intelligence. Morgan Kaufmann Publishers (2001)
4. García, L.J., Rupérez, D., Sandoval, A.L.: Bioinspired routing protocol for mobile ad hoc networks. IET Communication 4(18), 2187–2195 (2010)
5. Bullnheimer, B., Kostis, G., Strauss, C.: Parallelization strategies for the ant systems. In: High Performance Algorithms and Software in NonLinear Optimization Series: Applied Optimization, vol. 24 (1998)
6. Stützle, T.: Parallelization Strategies for ant Colony Optimization. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 722–731. Springer, Heidelberg (1998)
7. Thakur, R., Gropp, W.: Test suite for evaluating performance of multithreaded MPI communication. Parallel Computing 35(12), 608–617 (2009)
8. The NS-3 network simulator (2011), `http://www.nsnam.org`

ICARIS 2012

Conference Programme

The 11th International Conference on

Artificial Immune Systems

San Domenico Palace Hotel
Taormina, Italy
August 28-31, 2012

Friday August 31st - "Room Etna"

08:40-10:30 **Workshop** on *"Bio- & Immune- Inspired Algorithms and Models for Multi-Level Complex Systems"* – Chair: G. Franco

- "A computational-analysis of repeat sharing gene networks", **Giuditta Franco**, University of Verona, Italy
- "Stability-based Model Selection For High Throughput Genomic Data: An Algorithmic Paradigm", **Raffaele Giancarlo**, University of Palermo, Italy and **Filippo Utro** - Computational Biology Center, IBM T.J. Watson Research Center, USA
- "The immune system as a metaphor for topology driven patterns formation in complex systems", **Emanuela Merelli**, - University of Camerino, Italy and **Mario Rasetti** - ISI Foundation, Torino - Italy
- "Towards an evolutionary procedure for reverse-engineering biological networks", **Alberto Castellini, Vincenzo Manca and Mauro Zucchelli**, University of Verona, Italy
- "Distributed computing with prokaryotic immune systems", **Niall Murphy and Alfonso Rodriguez-Paton** - Universidad Politécnica de Madrid, Spain

10:30-11:00 **Coffee Break**

11:00-12:20 **Session IV** – Chair: J. Greensmith

- "An ecological approach to anomalies detection: the EIA model", **Pedro Pinacho, Iván Pau, Max Chacón and Sergio Sánchez**
- "Mathematical Implementation of Interation Between Malaria and Immune System", **Cicero Hildenberg Lima de Oliveira, Thayná Baptista Moroso, Fábio Hugo Souza Matos, Carolina Yukari Veludo Watanabe, Ciro José Egoavil Montero, Carlos Alberto Tenório de Carvalho Júnior, Hugo Fernando Maia Milan and Fernando Berton Zanchi**
- "Rethinking Concepts of the Dendritic Cell Algorithm for Multiple Data Stream Analysis", **Chris Musselle**
- "RC-DCA: A New Feature Selection and Signal Categorization Technique for the Dendritic Cell Algorithm Based on Rough Set Theory", **Zeineb Chelly and Zied Elouedi**

Friday August 31st cont... - "Room Etna"

12:20-13:30 **Oral Presentation** – Chair: G. Franco

- "Feature Subset Selection using GA-based Wrapper Approach and Clonal Selection Algorithms for Animal Breeding Data Mining" **Olgierd Unold, Henryk Maciejewski, Pawel Skrobanek, Ewa Walkowicz, Maciej Dobrowolski**
- "Immune Systems for ACO-Based Routing Optimization" **Delfín Rupérez Cañas, Luis Javier García Villalba**
- "Clustering T Cell Subsets Using Clonal Selection" **Stephanie J. Foan, Andrew M. Jackson, Ian Spendlove, Uwe Aickelin, Julie Greensmith**
- "The Dendritic Cell Algorithm: Review and Evolution", **Julie Greensmith**

13:30 **Concluding Remarks and Awards**

Immune Systems for ACO-Based Routing Optimization

Delfín Rupérez Cañas, Luis Javier García Villalba
Universidad Complutense de Madrid, Spain

Artificial Immune Systems (AIS) are used for solving complex optimization problems and can be applied to the detection of abnormal behaviours, such as fault tolerance. Optimization problems are a specific kind of problems, which we face, every day: to improve the efficiency of the resources of the devices, to find the shortest path between two points, to distribute resources among users uniformly. In general, an optimization problem is formulated as the minimization or maximization a function goal, which relates relevant variables of the problem domain finding the set of values that maximize or minimize such a function during the optimization process. The variables represent the problem domain, and the objective function characterizes the wanted goal. The variables are linked to restrictions to ensure acceptable solutions within the context. One of the optimization algorithms, based on the colony of ants [1], is cited in the literature frequently. It is inspired by the behaviour of ants at the time of obtaining food and in many areas is applied. The ACO algorithms consist of agents that operate without the need of a centralized control structure, in such a way local interactions from each agent and its neighbours allow communication among them in an autonomous way. These algorithms can be used to solve routing problems [2], being suitable for highly dynamic environments. As a starting point we present the routing protocol AntOR [3]. It is a hybrid ACO protocol based on AntHocNet [4]. Our protocol has the following characteristics:

- Disjoint-link and disjoint-node protocol [5].
- Separation between the pheromones values in the diffusion process.
- Use of the distance metric in the proactive path exploration.

AntOR provides two versions in its design: the disjoint-link (AntOR-DLR) in which the links are not shared and disjoint-node (AntOR-DNR) in which the nodes are not shared. Every disjoint-node is also a disjoint-link, but not vice versa. Both types of disjoint routes have the following advantages:

1. A failure in one node only affects a path, not the entire network.
2. Load balancing is better because there are not repeated routes on the disjoint property.

However, the use of such routes does also have its disadvantages, for example as the need for more resources by not sharing the links or nodes. As AntOR is still in research phase, we provide in this work a new variant to improve the overhead. To do this we use a new technique that makes the disjoint path exploration to not require the use of virtual pheromone. It also provides new mechanisms to improve the fault tolerance. To this end, we use a new optimization of the route local repair process. We are aiming at achieving with this work, better results in simulations according to the established performance metrics: Delivered information ratio, delays between sender-receiver (Average End-to-End delays), system overhead, Throughput and delays between consecutive data (Jitter).

References

[1] M. Dorigo and T. Stützle, “Ant Colony Optimization”, The MIT Press, 2004.

[2] M. Abolhasan, T. Wysocki and E. Dutkiewicz, “A review of routing protocols for mobile ad hoc networks”, Ad Hoc Networks, Vol. 2, No. 1, pp. 1-22, 2004.

[3] L.J. García Villalba, D. Rupérez Cañas and A.L. Sandoval Orozco, “Bioinspired routing protocol for mobile ad hoc networks”, IET Communications, Vol. 4, No. 18, pp. 2187-2195, 2010.

[4] F. Ducatelle, Adaptive routing in ad hoc wireless multi-hop networks, PhD thesis, Università della Svizzera Italiana, Istituto Dalle Molle di Studi sull´Intelligenza Artificiale, 2007.

[5] D. Rupérez Cañas, A.L. Sandoval Orozco, L.J. García Villalba and T.H. Kim, “A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks,” in Communications in Computer and Information Science (CCIS), Vol. 263, pp. 300-304, 2011.

XXVII Simposium Nacional de la Unión Científica Internacional de Radio

Elche, 12 - 14 de septiembre de 2012

UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

PROGRAMA – LIBRO DE RESÚMENES

Elche, del 12 al 14 de septiembre de 2012

Organizadores:
Departamento de Ingeniería de Comunicaciones
Escuela Politécnica Superior de Elche
Universidad Miguel Hernández de Elche

Editores:
Germán Torregrosa Penalva
Enrique Bronchalo Bronchalo
Adrián J. Torregrosa Fuentes
Javier Gozálvez Sempere
Ángel A. San Blas Oltra
Juan Capmany Francoy

Depósito Legal: MU 717-2012
ISBN: 978-84-695-4326-9

ANTOR-UDLR: Aproximación Unicast de un Protocolo de Encaminamiento para Redes Móviles Ad Hoc

Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba
{delfinrc, asandoval, javiergv}@fdi.ucm.es
Grupo de Análisis, Seguridad y Sistemas (GASS)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid

***Abstract*— A mobile ad hoc network is a collection of mobile devices that forms a communication network without predefined infrastructure. This determines that the design of routing protocols for such networks is a complex task. A particular type of routing protocols are called bioinspired, which take into account the behavior of some animals at the time of obtaining their food. A representative bioinspired protocol is AntOR. A protocol called AntOR-UDLR is proposed in this work, which consists in a unicast approach of bioinspired protocol AntOR-DLR. This approach consists of replacing the notification messages of link failure, which are sent in broadcast mode in AntOR-DLR, by unicast messages that are sent to the predecessor of the node reporting the link failure until they reach the source of the session data. The simulation results show that this new protocol improves on its predecessors in all metrics analyzed.**

I. INTRODUCCIÓN

Las redes móviles ad hoc (MANETs) [1] están formadas por dispositivos móviles inalámbricos que se comunican de forma distribuida. Consecuentemente, el diseño de protocolos de encaminamiento eficientes es un aspecto fundamental [2] [3] [4] [5] [6] [7] [8]. Un tipo particular de protocolos que centran muchas investigaciones son los denominados bioinspirados. Muchos protocolos bioinspirados han sido propuestos en la literatura, siendo especialmente representativos los basados en el Problema de Optimización de la Colonia de Hormigas [9] o, abreviadamente, ACO, correspondiente al acrónimo de su terminología inglesa (Ant Colony Optimization). Así, P. Deepalakshmi y S. Radhakrishnan [10] introducen un proceso de reenvío probabilístico para satisfacer las calidades de servicio que son adaptadas automáticamente a la movilidad de los nodos en las MANETs en protocolos multicast. J. Jain et al. [11] usan un método donde ACO es usado en el caso de fallo de enlace. Los autores afirman que pueden mejorarse el throughtput y el ratio extremo-a-extremo. Probablemente la sobrecarga se reduce debido a que los paquetes de control son hormigas hacia adelante y hormigas hacia atrás. Pero, sin duda, el protocolo más representativo es AntHocNet [12] [13] [14], protocolo adaptativo y multicamino que tiene en cuenta la topología dinámica y otras características de las MANETs y que presenta un funcionamiento híbrido: es reactivo porque tiene agentes que operan bajo demanda para establecer rutas a los destinos y es proactivo porque tiene otros agentes que obtienen información para descubrir nuevas rutas alternativas en la prevención ante los fallos de enlace. Finalmente, en [15] se presenta AntOR, un protocolo ACO híbrido basado en AntHocNet, que tiene las siguientes características que lo diferencian de AntHocNet:

- Protocolo de ruta disjunta de enlace y de nodo.
- Separación entre los valores de las feromonas en el proceso de difusión.
- Uso de la métrica distancia en la exploración proactiva de rutas.

AntOR contempla dos versiones en su diseño: La versión disjunto de enlace (AntOR-DLR) es aquella en la que los enlaces no se comparten. Por otro lado, en la versión disjunta de nodo (AntOR-DNR) son los nodos los que no se comparten. Toda ruta disjunta de nodo es disjunta de enlace, pero no al revés. Los dos tipos de rutas tienen las siguientes ventajas:

a. Cuando falla un nodo afecta a un camino, pero no a toda la red.
b. Con la propiedad disjunta el balanceo de la carga es mejor, porque no se repiten rutas.

Aunque tiene algunas desventajas como la necesidad de más recursos al no compartir los enlaces ni los nodos.

En este trabajo se presenta AntOR-UDLR (AntOR- Unicast Disjoint Link Route), variante unicast de AntOR-DLR (AntOR- Disjoint Link Route). Se ha elegido AntOR-DLR frente a AntOR-DNR por la comparativa realizada [16] que demuestra que el primero posee mejores prestaciones. La idea principal de AntOR-UDLR es sustituir los mensajes de notificación de fallo de enlace enviados en modo broadcast por mensajes unicast enviados al nodo predecesor de una ruta válida a un destino alcanzable.

Este artículo se compone de 4 secciones, siendo la primera la presente introducción. En la sección 2 se presenta AntOR-UDLR explicándose las principales diferencias respecto a AntOR-DLR. En la sección 3 se muestran los resultados de la comparativa realizada entre AntOR-UDLR y AntOR-DLR.

Finalmente, las conclusiones y el trabajo futuro son expuestos en la sección 4.

II. ANTOR-UDLR

En este trabajo presentamos un nuevo protocolo que es una variante de AntOR-DLR. La idea principal de esta aproximación es sustituir los mensajes de notificación enviados en modo broadcast por mensajes sencillos enviados al precursor de una ruta válida a un destino alcanzable. Se entiende por ruta válida a aquella que tiene feromona mayor que cero y pertenece a la sesión activa de un determinado destino. En el diseño de AntOR-UDLR se ha tenido en cuenta el mensaje unicast de notificación de enlace (ULN). Este contiene dos direcciones IP: Session Destination Address y Session Source Address. Para entender esta propuesta es necesario diferenciar la forma de administración de los fallos de enlace entre AntOR-DLR y AntOR-UDLR.

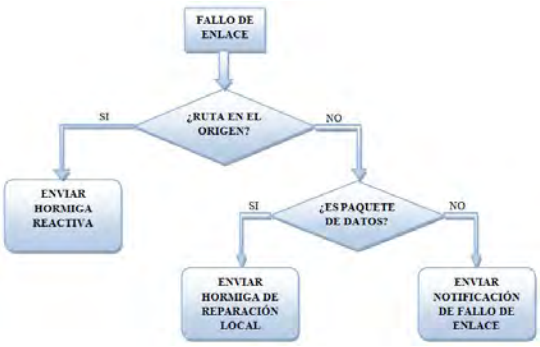


Fig. 1. Administración de fallos de enlace en AntOR-DLR.

Lo primero que ocurre cuando hay un fallo de nodo es que el nodo en AntOR-DLR que percibe el fallo elimina a este de su tabla de rutas. A continuación, se actualiza la tabla de encaminamiento con la nueva información de feromona. Por último se procesa dependiendo de varios factores:

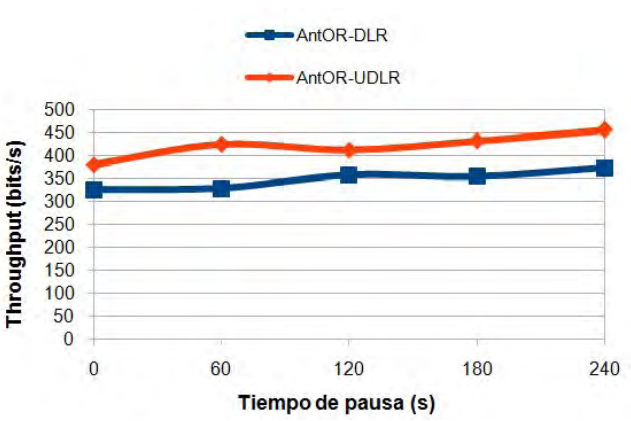
a. Si no hay ruta en el origen se envía una hormiga reactiva hacia adelante.
b. Si no hay ruta en un nodo intermedio y se trata de un paquete de datos lo que se estaba retransmitiendo cuando se produjo el fallo, se envía una hormiga de reparación de ruta hacia adelante. Si no hay respuesta de la correspondiente hormiga de reparación hacia atrás en un determinado periodo de tiempo se envía en modo broadcast un mensaje de notificación de fallo de enlace informando que el destino es inalcanzable.
c. Si no hay ruta en un nodo intermedio y se trata de un paquete de control, que puede ser un mensaje HELLO no recibido consecutivamente cada cierto intervalo o un mensaje de control unicast, se crea un mensaje de notificación de fallo de enlace informando de los destinos inalcanzables y se envía en modo broadcast. En la Fig. 1 podemos observar el proceso de notificación de fallos en AntOR-DLR.

Como en AntOR-DLR, lo primero que ocurre cuando hay un fallo de nodo en AntOR-UDLR es que el nodo que percibe

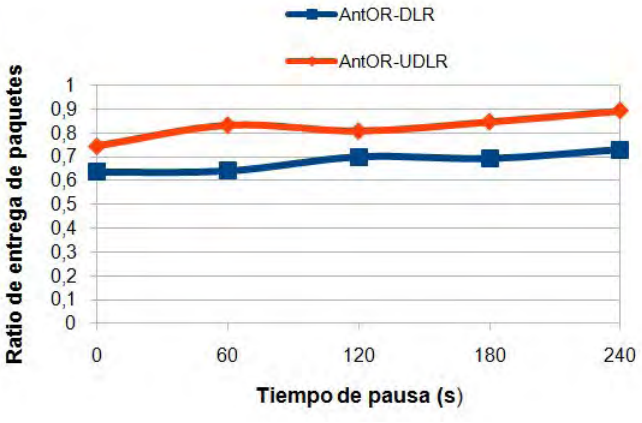
Fig. 2. Administración de fallos de enlace en AntOR-UDLR.

el fallo elimina a este de su tabla de rutas. A continuación, se actualiza la tabla de de encaminamiento con la nueva información de feromona. Por último, se procesa de manera parecida a AntOR-DLR:

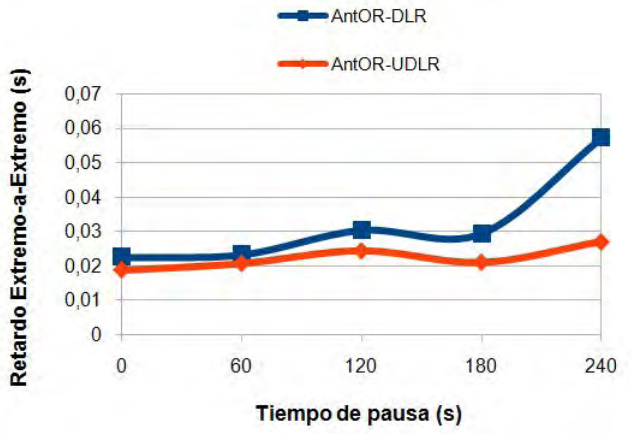
a. Si no hay ruta en el origen se envía una hormiga reactiva hacia adelante.
b. Si no hay ruta en un nodo intermedio, y se trata de un paquete de datos lo que se estaba retransmitiendo cuando se produjo el fallo, se envía una hormiga de reparación de ruta hacia adelante. Si no hay respuesta de la correspondiente hormiga de reparación hacia atrás en un determinado periodo de tiempo se envía un mensaje en modo unicast al precursor de la ruta informando que el destino es inalcanzable. El nodo que recibe este mensaje actualiza la tabla de encaminamiento y reenvía este mensaje al precursor de la ruta al destino. Este proceso se repite tantas veces hasta que se llega al nodo origen de la sesión de datos.
c. Si no hay ruta en el nodo intermedio, y si se trata de un paquete de control (un mensaje HELLO no recibido consecutivamente cada cierto intervalo o un mensaje de control unicast) no se envía ningún mensaje. Esto último puede originar que haya rutas que no se reparen correctamente, porque no se realiza ninguna operación cuando se trata de un paquete de datos. Por eso, existe una funcionalidad nueva: Cuando un nodo intermedio, que encamina los datos, no encuentra una ruta válida (con feromona regular mayor que cero) envía un mensaje de unicast a todos los vecinos a un salto, para que actualicen sus tablas de encaminamiento. Es necesario enviar este mensaje a todos los vecinos, porque al no encontrar ruta válida no tenemos información del precursor. Cuando uno de estos nodos vecinos tiene una ruta válida al destino reenvía el mensaje unicast al precursor de la ruta. Este proceso se repite tantas veces hasta que se llega al nodo origen. En la Fig. 2 observamos el proceso de notificación de fallos en AntOR-UDLR.



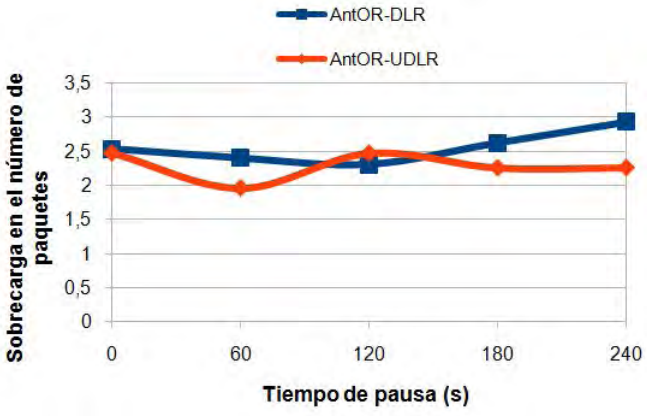
(a) *Throughput.*



(b) *Ratio de entrega de paquetes de datos.*



(c) *Retardo extremo-a-extremo.*



(d) *Sobrecarga en el número de paquetes.*

Fig. 3. Métricas analizadas.

III. AntOR-UDLR frente a AntOR-DLR

Para analizar el funcionamiento de este nuevo protocolo se analizaron las siguientes métricas:

- Throughput: Volumen de trabajo o información que fluye a través del sistema. Se calcula dividiendo el número total de bit entregados al destino por el tiempo de entrega de paquetes.
- Ratio de paquetes de datos entregados: Relación entre el número de paquetes enviados y el número de paquetes entregados correctamente.
- Retardo medio Extremo-a-Extremo: Medida de la efectividad acumulativa de los retardos experimentados por los paquetes que van desde el origen al destino.
- Sobrecarga en el número de paquetes: Relación entre el número total de paquetes de control transmitidos y el número de paquetes de datos entregados.

La experimentación se hizo en el simulador de Red NS-3 (versión 3.12.1). Las características de las simulaciones fueron: Se utilizaron 100 nodos distribuidos aleatoriamente y configurados según el estándar IEEE 802.11b, con ancho de banda de 2 MBit/s y con un solo canal de comunicación y rango de transmisión de 300 m. Los nodos se movían según el patrón *Random WaitPoint* (RWP) variando el tiempo de pausa de un mínimo de 0 s hasta 240 s en intervalos de 60 s. El escenario era rectangular de dimensiones 3000 m $\times$ 1000 m. La velocidad era variable de un mínimo de 0 m/s hasta 2.5 m/s. Se usaron 10 sesiones de datos aleatorias utilizando el protocolo de aplicación *Constant Bit Rate* (CBR) que empezaba a enviar datos de forma aleatoria desde 0 s hasta un máximo de 60 s. La tasa de envío fue de 512 bit/s, es decir, un envío de un paquete de 64 bytes por segundo. El tiempo máximo de simulación se estableciόe a 300 s. Se emplearon un total 5 ejecuciones en el experimento.

En la Fig. 3(a) observamos como el Throughput es mayor en AntOR-UDLR que en AntOR-DLR para cualquier tiempo de pausa. Tanto para escenarios muy dinámicos (tiempo de pausa 0 s) como escenarios donde se valora la pérdida de conectividad (tiempo de pausa 240 s).

En la Fig. 3(b) vemos un comportamiento parecido con respecto a la Fig. 3(a), pero utilizando otra escala. Donde observamos El ratio es una métrica muy importante que nos da información sobre la eficiencia del protocolo.Se observa que el ratio en AntOR-UDLR nunca es inferior al 70 % sea cual sea el tiempo de pausa. Asimismo, se observa cómo en nuestra

aproximación se consigue entregar con éxito más paquetes de datos a los destinos. En este escenario y con este nuevo algoritmo hemos conseguido alcanzar un ratio de 90 % en unas condiciones adversas porque es un escenario rectangular que no favorece la recepción de paquetes. Los escenarios cuadrados tienen una mejor distribución de los nodos y un movimiento más regular y uniforme.

En la Fig. 3(c) vemos como el retardo extremo-a-extremo es menor en AntOR-UDLR que en AntOR-DLR en todo momento. También podemos comprobar que la curva que representa el retardo en la versión unicast es más uniforme que en la versión original disjunto de enlace. Este comportamiento nos hace pensar que AntOR-UDLR tiene una condición de equilibrio más estable que AntOR-DLR ante un diferente patrón de movilidad.

En la Fig. 3(d) observamos cómo la sobrecarga en el número de paquetes es inferior en AntOR-UDLR que en AntOR-DLR, exceptuando un punto como consecuencia de las ejecuciones aleatorias. Si hiciésemos un número mayor de pruebas, conseguiríamos que la media fuese más exacta. Nosotros hemos tomado un intervalo de confianza del 95 %. En este punto de tiempo de pausa (120 s), los intervalos de confianza son para AntOR-DLR [1,468 - 3,1432] y para AntOR-UDLR [0,887 - 4,048]. En AntOR-UDLR podemos estar seguros en un 95 % de que el valor medio en este punto oscila entre [0,887 - 4,048]. Si aumentamos el número de ejecuciones reducimos el intervalo de confianza, porque se va acotando la distancia que limita la media. Por ejemplo, en AntOR-UDLR para 20 ejecuciones para el mismo intervalo de confianza, obtenemos la media comprendida entre [1,677 - 3,258]. Como vemos, a medida que disminuye el intervalo, conseguimos que el error sea menor. En un escenario en el que la conectividad se pierde, debido a un mayor tiempo de pausa, ocasiona pares de origen-destino sin rutas para transmitir los datos. En este caso AntOR-UDLR tiene una sobrecarga más baja que AntOR-DLR porque se envían paquetes unicast en vez de broadcast.

IV. Conclusiones y Trabajo Futuro

En este trabajo se ha presentado una nueva aproximación, AntOR-UDLR, del protocolo encaminamiento AntOR-DLR. La idea principal de esta variante es sustituir, en el proceso de fallos de enlace, los mensajes de notificación de fallos enviados en modo broadcast por mensajes unicast enviados al predecesor en el destino de una ruta valida. Los resultados muestran que AntOR-UDLR mejora a su AntOR-DLR en las métricas de retardo medio extremo-a-extremo, sobrecarga en el número de paquetes, Ratio de entrega de paquetes y Throughput.

Como trabajo futuro se baraja sustituir los mensajes de reparación local de ruta por mensaje unicast así como profundizar en la experimentación y evaluar más métricas como podría ser el Jitter. Además, en la futura experimentación pretendemos utilizar otra clase de escenarios (p.e. de dimensiones cuadradas), en los cuales se pueden utilizar otros modelos de movilidad. También se pueden variar los parámetros internos de configuración del protocolo para mejorar los resultados.

Agradecimientos

Los autores agradecen la financiación que les brinda el Subprograma AVANZA COMPETITIVIDAD I+D+I del Ministerio de Industria, Turismo y Comercio (MITyC) a través del Proyecto TSI-020100-2011-165. Asimismo, los autores agradecen la financiación que les brinda el Programa de Cooperación Interuniversitaria de la Agencia Española de Cooperación Internacional para el Desarrollo (AECID), Programa PCI-AECID, a través de la Acción Integrada MAEC-AECID MEDITERRÁNEO A1/037528/11.

References

[1] M. Abolhasan, T. Wysocki and E. Dutkiewicz, “A review of routing protocols for mobile ad hoc networks,” *Ad Hoc Networks,* Vol. 2, No. 1, pp. 1-22, 2004.

[2] T. Clausen and P. Jacquet,“Optimized link state routing protocol (OLSR),” *IETF RFC3626*, October 2003. www.ietf.org/rfc/rfc3626.txt

[3] C.E. Perkins, E.M. Belding-Royer and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” *IETF RFC3561*, July 2003. http://tools.ietf.org/html/rfc3561

[4] D. Jonson, Y. Hu and D. Maltz, “The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4,” *IETF RFC4728*, February 2007. http://www.ietf.org/rfc/rfc4728

[5] R. Ogier, F. Templin and M. Lewis, “Topology Dissemination Based on Reverse-Path Forwarding (TBRPF), ” *IETF RFC3684*, February 2004. http://www.ietf.org/rfc/rfc3684

[6] Z.J. Haas and M.R. Pearlman, “The zone routing protocol (ZRP) for ad hoc networks, ” *IETF Draft*, July 2002. http://tools.ietf.org/id/draft-ietf-manet-zone-zrp-04.txt

[7] M. Gerla, X. Hong and G. Pei, “Fisheye State Routing Protocol (FSR) for Ad Hoc Networks, ” *Internet Draft*, June 2002. http://tools.ietf.org/html/draft-ietf-manet-fsr-03

[8] V. Park and S. Corson, “Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Specification,” *Internet Draft*, July 2001. http://tools.ietf.org/html/draft-ietf-manet-tora-spec-04

[9] M. Dorigo and T. Stützle, “Ant Colony Optimization,” *The MIT Press*, 2004.

[10] P. Deepalakshmi and S. Radhakrishnan, “Source-Initiated QoS Multicasting Scheme for MANETs Using ACO,” *International Conference on Process Automation, Control and Computing (PACC)*, pp. 1-4, July 2011.

[11] J. Jain, R. Gupta and T.K. Bandhopadhyay, “Ant colony algorithm in MANET-local link repairing of AODV,” *3rd International Conference on Electronics Computer Technology (ICECT)*, Vol. 6, pp. 270-273, April 2011.

[12] F. Ducatelle, *Adaptive routing in ad hoc wireless multi-hop networks*, PhD thesis, Universita‘ della Svizzera Italiana, Istituto Dalle Molle di Studi sull´Intelligenza Artificiale, 2007.

[13] G.A. Di Caro, F Ducatelle and L.M. Gambardella, “AntHocNet: An Adaptive Nature-Inspired Algorithm for Routing in Mobile Ad Hoc Networks,” *European Transactions on Telecommunications*, Special Issue on Self-organization in Mobile Networking, Vol. 16, Issue 5, October 2005.

[14] G. Di Caro, F. Ducatelle and L.M. Gambardella, “AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks,” *Proceedings of PPSN VIII - Eight International Conference on Parallel Problem Solving from Nature*, Birmingham, UK, Springer-Verlag, Lecture Notes in Computer Science, Vol. 3242, September 18-22, 2004.

[15] L.J. García Villalba, D. Rupérez Cañas and A.L. Sandoval Orozco, “Bioinspired routing protocol for mobile ad hoc networks,” *IET Communications*, Vol. 4, No. 18, pp. 2187-2195, 2010.

[16] D. Rupérez Cañas, A.L. Sandoval Orozco, L.J. García Villalba and T.H. Kim, “A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks,” in *Communications in Computer and Information Science (CCIS)*, Vol. 263, pp. 300-304, 2011.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 532572, 5 pages
doi:10.1155/2012/532572

Research Article

Multiple Interface Parallel Approach of Bioinspired Routing Protocol for Mobile Ad Hoc Networks

L. J. García Villalba,[1] D. Rupérez Cañas,[1] A. L. Sandoval Orozco,[1] and T.-H. Kim[2]

[1] *Grupo de Análisis, Seguridad y Sistemas (GASS), Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA), Facultad de Informática, Universidad Complutense de Madrid (UCM), Despacho 431, Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain*
[2] *School of Information Science, GVSA and UTAS, 20 Virginia Court, Sandy Bay, Hobart, TAS 7001, Australia*

Correspondence should be addressed to L. J. García Villalba, javiergv@fdi.ucm.es

Received 30 September 2012; Accepted 6 October 2012

Academic Editor: Sabah Mohammed

The design of routing protocols for mobile ad hoc networks (MANETs) is a complex task given the dynamic nature of such networks. Particular types of routing protocols are known as bioinspired. This work presents a parallelization of AntOR-DNR, a bioinspired routing protocol for mobile ad hoc networks based on the Ant Colony Optimization (ACO) algorithm. This new protocol, called PantOR-MI, uses, as well as PAntOR, the thread programming based on shared memory. This new parallelization is applied in route discovery phases, route local repair process, and link failure notification. The simulation results indicate that PAntOR and PAntOR-MI improve performances of AntOR, whilst it is also noticed that PAntOR-MI is the most suitable for highly dynamic environments.

1. Introduction

A mobile ad hoc network (MANET) [1] is a collection of mobile devices, which form a network of communication without predefined infrastructure. This fact determines the design of routing protocols for this type of network to suppose an arduous task. Particular types of routing protocols are called bioinspired [2], taking into account the behavior of some animals (insects, etc.) to obtain their food. Related to theses, the algorithms based on Ant Colony Optimization (ACO) [3], are particularly relevant. A representative protocol of so-called bioinspired is AntOR [4], adaptive and multihop routing protocol based on AntHocNet [5, 6]. The specification of this protocol includes two versions: disjoint-link routes (AntOR-DLR) and disjoint-node routes (AntOR-DNR). A parallel approximation of AntOR-DNR is PAntOR [7, 8] improving performances of AntOR-DNR through thread programming based on shared memory in the phases of routing information setup, route local repair, and link failure notification. This paper presents a new parallel approximation of AntOR-DNR, called PAntOR-Multiple Interface (PAntOR-MI) which, as its name suggests, differs essentially from PAntOR in the use of multiple interfaces. This paper consists of 6 sections, with this introduction being the first of them. The rest of the paper is structured as follows: Section 2 briefly discusses the most representative work on parallel techniques for bioinspired protocols based on the behavior of ants. Section 3 briefly comments on AntOR-DNR and PAntOR, also showing a comparison between both. Section 4 presents PAntOR-MI, with emphasis on differences from its predecessor, PAntOR. Section 5 shows a comparative study between AntOR-DNR, PAntOR, and PAntOR-MI. Finally, the conclusions are exposed in Section 6.

2. Related Works

In this section we present the most representative parallelization techniques that make ACO more efficient. First of which, introduced by [9], explains a method that can solve difficult combinatorial optimization problems. Stützle [10] applies an approximation master/slave to parallelize several different

search methods of ACO solutions. Reference [11] shows a hybrid system of parallelization which consists of evaluating the performance of communication multithreading Message Passing Interface (MPI). This approach applies MPI between nodes and multithreading within nodes. Finally, [7, 8] presents PAntOR, an approximation parallel based on programming by threads, which constitutes the starting point of the present paper. The main idea of this protocol is to replace each broadcast message by a message managed by a thread that is addressed to each one-hop neighbour, that is, launches a thread by each node in the neighbour table. This is done in protocol phases: routing information setup, route local repair, and link failure notification.

3. AntOR versus PAntOR

AntOR [4] has the following characteristics which are different from AntHocNet [5, 6]:

(i) Disjoint-link and Disjoint-node protocol,

(ii) separation between the pheromone values in the diffusion process, and

(iii) use distance metric in route proactive exploration.

PAntOR [7, 8] paralyze Disjoint-node version (AntOR-DNR) that consists of nodes from routes which are not shared. We have chosen this version because routes are more difficult to get and maintain. To understand how P-AntOR works, it is necessary to use three concepts well.

(a) *Process*. It is a program running which is managed by the Operating System.

(b) *Thread*. It consists of the basic unit of execution, so any program that executes has at least a thread.

(c) *POSIX Thread*. It is a Standard based in thread API for C/C++.

We use POSIX Thread because it allows a new concurrent process flow to expand, which is the most efficient multicore systems, generating a flow of processes that can be scheduled to run on another processor, thus speed through distributed processing is achieved. Programming with threads carries less overhead than expanding a new process, because the system does not initialize a new environment and virtual memory space for that process.

This version tries to replace each broadcast message by a message managed by a thread that is addressed to each one-hop neighbour.

This parallel technique, which launches a thread by each node in the neighbour table, is used in the following phases of AntOR.

3.1. Routing Information Setup. Figure 1 shows a flow chart representing the parallelism in the route discovery process. When a source node is ready to send data to the destination node, it activates the route discovery process. This process is parallelized using threads, so that it launches a Reactive Forward Ant (RFA) reactive to the one-hop neighbours

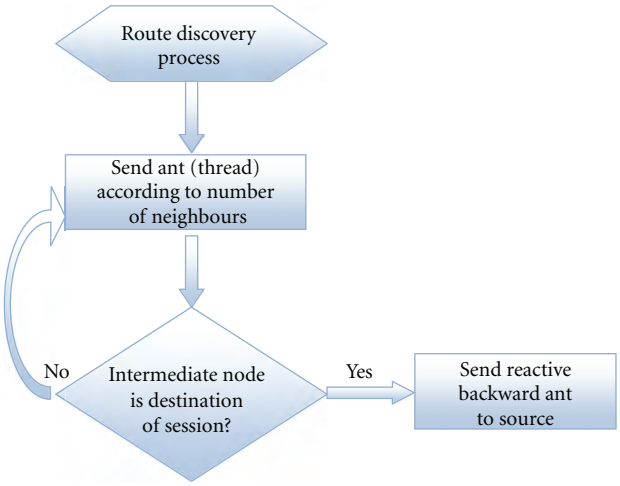


Figure 1: Parallelization of route discovery phase.

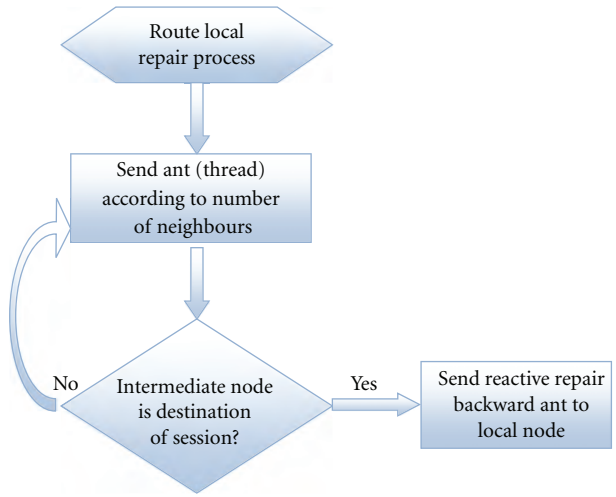


Figure 2: Parallelization of local route repair phase.

through an independent thread. When an intermediate node receives this ant repeats the process, but whether it is a destination node, this node sends its corresponding Reactive Backward Ant (RBA).

3.2. Route Local Repair. The operation is similar to route discovery, unless it is done locally, as shown in Figure 2.

3.3. Link Failure Notification. This process updates the routing table to link failures. It is required to be taken promptly because of its importance. The nodes send ants through independent threads until an intermediate node has some alternative route to the destination after updating the routing table, as shown Figure 3.

3.4. Results. Next we show a comparison between PAntOR and AntOR. To this end, the metrics used in the simulations were as follows.

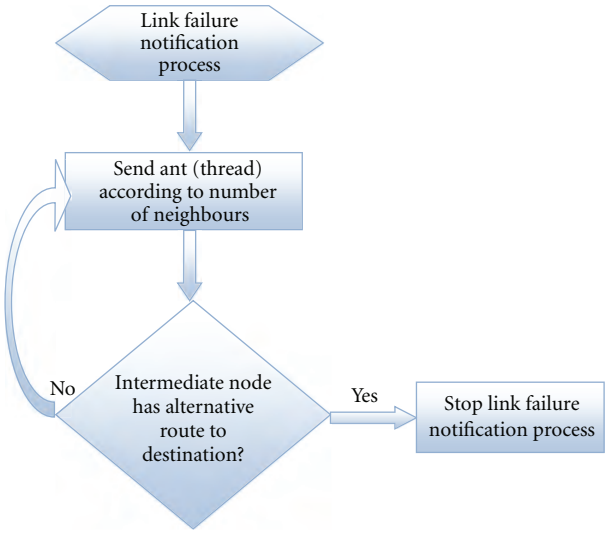


Figure 3: Parallelization of link failure notification phase.

(i) *Throughput*: consists of volume of work or information flowing through a system. It is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.

(ii) *Delivered Data Packet Ratio*: relationship between number of packets sent and the number of packets delivered successfully.

(iii) *Overhead in number of packets*: relationship between the total numbers of transmitted control packets by the nodes of network and the number of delivered data packets to their destinations.

Experiments with the Network Simulator NS-3 have been realized. Simulations parameters are as follows: we have used 100 nodes configured according to the Standard IEEE 802.11b, moving in a random scenario of dimensions 1200 m × 1200 m according to the mobility pattern *Random WayPoint* (RWP). The application of data traffic is *Constant Bit Rate* (CBR) with a rate of packet sending of 2048 bps (4 packets of 64 bytes per second). We apply 5 random data sessions, where mobility is variable from 0 m/s up to a maximum of 10 m/s. Pause time is kept constant to a value of 30 s. Total simulation time is 120 s. In Figure 4 we can observe how the Throughput is better at PAntOR at AntOR.

In Figure 5 see how overhead in number of packets is better in the parallel version than in the original, because the creation of routes is faster.

Figure 6 behaves similar to the representation in Figure 4, but using another scale. It notes that the performance of the packet delivery remains reasonably good even at high speeds.

4. PAntOR-Multiinterface

This variant of PAntOR consists of having more than one interface, and to parallelize the sending of broadcast messages by interface through threads. Each interface is managed by a thread. The main idea of this parallelization

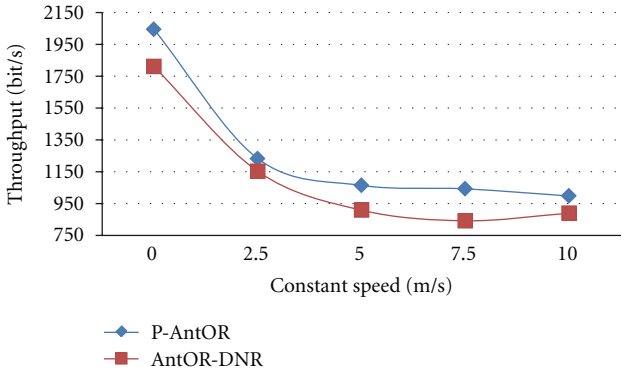


Figure 4: Constant speed against Throughput.

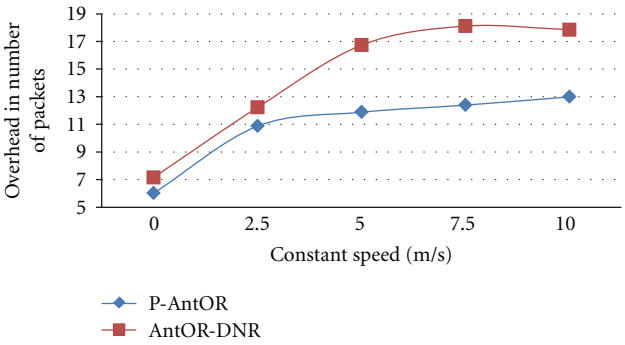


Figure 5: Constant speed against overhead in number of packets.

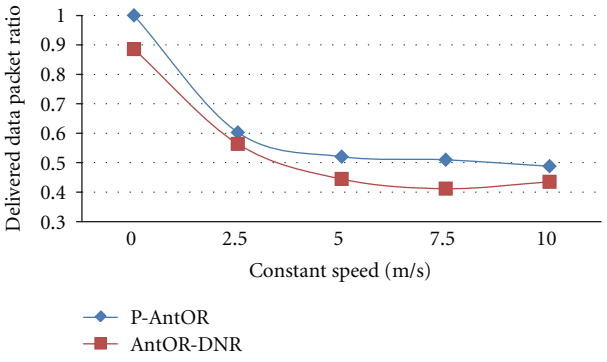


Figure 6: Constant speed against Delivered Data Packet Ratio.

is to be applied to systems with several multiinterfaces, which launches an ant broadcast mode in an independent thread for each interface that provides the node. The main difference with PAntOR is that PAntOR-MI uses more than one interface, parallelizing each interface by means of a thread.

To understand this variant we provide Algorithm 1.

It can be seen that while running the routing information setup, a reactive message is launched in a broadcast way for each interface that have the node, and such an interface via a thread is managed.

```
while Routing Information Setup do
    for Cont = 1 to Max_Interfaces do
        Send Broadcast Message by Thread (Cont);
    end
end
```

Algorithm 1: Parallelization core in PAntOR-MI.

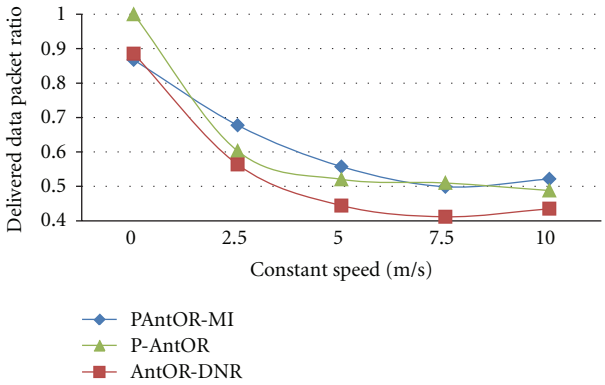


Figure 7: Constant Speed against Delivered Data Packet Ratio.

5. PAntOR-MI versus AntOR and PAntOR

The performance metric analyzed in this comparison is Delivered Data Packet Ratio, which consists of the relationship between the number of packets sent and the number of packets delivered correctly. To perform this comparison Network Simulator NS-3 has been used with the following parameters: a random scenario with dimensions of 1200 m $\times$ 1200 m has been designed, where 100 nodes configured according to the Standard IEEE 802.11b; they move according to the mobility pattern *Random WayPoint* (RWP). The application of data traffic is based on *Constant Bit Rate* (CBR) with a packet sending rate of 2048 bps (4 packets of 64 bytes per second). We apply 5 random data sessions, where mobility is variable from 0 m/s up to a maximum of 10 m/s. Pause time is kept constant to a value of 30 s. Total simulation time is 120 s. In this comparison PAntOR-MI use nodes with two interfaces.

In Figure 7 we appreciate how the delivered packet ratio is better in these two parallel versions than in AntOR. Also we see how PAntOR-MI improves to AntOR and PAntOR. Also a greater tolerance to the mobility of the nodes is observed, behaving better in the more dynamic scenarios.

6. Conclusions

This work has presented a new bioinspired routing protocol for mobile ad hoc networks obtained thanks to new parallelization techniques of a base protocol called AntOR which has two versions, the so-called Disjoint-link (AntOR-DLR) and Disjoint-node (AntOR-DNR). The new parallel approach (PAntOR-MI) used the disjoint-node version of AntOR (AntOR-DNR) as the main protocol, as well as the existing (PAntOR). The parallelization technique employed is a large-grained approach, in which a multicore machine in a shared memory system has been used. The novelty of PAntOR-MI is that we have more than one interface, and we parallelize the sending of broadcast messages by interface using threads. Each interface is managed by a thread. The obtained simulation results indicate that PAntOR-MI improves performances of PAntOR, with further observations showing that this improvement is most evident in most dynamic environments.

Acknowledgment

This work was supported by the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID Mediterráneo A1/037528/11.

References

[1] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile Ad Hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, 2004.

[2] V. Jha, K. Khetarpal, and M. Sharma, "A survey of nature inspired routing algorithms for MANETs," in *Proceedings of the 3rd International Conference on Electronics Computer Technology (ICECT '11)*, pp. 16–24, Kanyakumari, India, April 2011.

[3] M. Dorigo and T. Stützle, *Ant Colony Optimization*, The MIT Press, Bradford Company Scituate, Mass, USA, 2004.

[4] L. J. G. Villalba, D. R. Cañas, and A. L. S. Orozco, "Bio-inspired routing protocol for mobile Ad Hoc networks," *IET Communications*, vol. 4, no. 18, pp. 2187–2195, 2010.

[5] F. Ducatelle, *Adaptive routing in Ad Hoc wireless multi-hop networks [Ph.D. thesis]*, Università della Svizzera Italiana, Istituto Dalle Molle di Studi Sull'Intelligenza Artificiale, 2007.

[6] G. Di Caro, F. Ducatelle, and L. M. Gambardella, "AntHocNet: an adaptive nature-inspired algorithm for routing in mobile Ad Hoc networks," *European Transactions on Telecommunications*, vol. 16, no. 5, pp. 443–455, 2005.

[7] L. J. García Villalba, D. Rupérez Cañas, and A. L. Sandoval Orozco, "Parallel approach of a bioinspired routing protocol for MANETs," *International Journal of Ad Hoc and Ubiquitous Computing*. In press.

[8] D. Rupérez Cañas, A. L. Sandoval Orozco, L. J. García Villalba, and T.-H. Kim, "Comparing AntOR-disjoint node routing protocol with its parallel extension," in *Proceedings of the International Conference on Multimedia, Computer Graphics and Broadcasting (MulGraB '11)*, pp. 305–309, 2011.

[9] B. Bullnheimer, G. Kostis, and Strauss, "Parallelization strategies for the ant systems," in *High Performance Algorithms and*

Software in NonLinear Optimization Series: Applied Optimization, vol. 24, 1998.

[10] T. Stützle, "Parallelization strategies for ant colony optimization," in *Proceedings of the Parallel Problem Solving from Nature*, vol. 1498, 1998.

[11] R. Thakur and W. Gropp, "Test suite for evaluating performance of multithreaded MPI communication," *Parallel Computing*, vol. 35, no. 12, pp. 608–617, 2009.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 956146, 5 pages
doi:10.1155/2012/956146

Research Article

Restrictive Disjoint-Link-Based Bioinspired Routing Protocol for Mobile Ad Hoc Networks

L. J. García Villalba,[1] D. Rupérez Cañas,[1] A. L. Sandoval Orozco,[1] and T.-H. Kim[2]

[1] *Grupo de Análisis, Seguridad y Sistemas (GASS), Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA), Facultad de Informática, Universidad Complutense de Madrid (UCM), Despacho 431, Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain*
[2] *School of Information Science, GVSA and UTAS, 20 Virginia Court, Sandy Bay, Hobart, TAS 7001, Australia*

Correspondence should be addressed to L. J. García Villalba, javiergv@fdi.ucm.es

Received 29 September 2012; Accepted 8 October 2012

Academic Editor: Sabah Mohammed

The design of routing protocols for mobile ad hoc networks (MANETs) is a complex task given the dynamic nature of such networks. Particular types of routing protocols are known as bioinspired. Related to theses, the algorithms based on Ant Colony Optimization (ACO), are particularly relevant. This work presents a new variant of AntOR, a multihop adaptive routing protocol based on AntHocNet which already has two versions: disjoint link routes (AntOR-DLR) and disjoint node (AntOR-DNR). The new protocol, called AntOR-RDLR, differs from AntOR-DLR in the pheromones updating process and the route discovery mechanism. The simulation results indicate that AntOR-RDLR improves their predecessors in all analyzed metrics.

1. Introduction

A mobile ad hoc network (MANET) [1] is a collection of mobile devices, which form a communication network without predefined infrastructure. This fact determines the design of routing protocols for this type of network to suppose an arduous task. Particular types of routing protocols are called bioinspired, which take into account the behaviour of some animals (insects, etc.) to obtain their food. A representative protocol of so-called bioinspired is AntOR [2], multihop adaptive routing protocol based on AntHocNet [3]. The specification of this protocol includes two versions: disjoint link routes (AntOR-DLR) and disjoint node (AntOR-DNR). This work presents a variant of the first one. This paper consists of 6 sections, with this being the first of them. The rest of the paper is structured as follows. Section 2 discusses briefly the most representative works in the area of bioinspired algorithms for their application in the design of routing protocols for mobile ad hoc networks. Section 3 presents AntOR, predecessor algorithm and its two versions; also making a comparison between both, whose analysis lays the main keys for AntOR-RDLR. Section 4 introduces AntOR-RDLR, emphasizing the differences with respect to its predecessor. A comparative study between AntOR-RDLR and AntOR-DLR is shown in Section 5. Finally, the conclusions are established in Section 6.

2. Related Work

Many bioinspired protocols have been proposed in literature. In ant routing algorithm for mobile ad hoc networks (ARAMAs) [4] discovery and route maintenance overhead is reduced through the control of the number of forward ants. However they do not clarify how to control the generation of ants in a dynamic environment. [5] presents a protocol that has a low delivered data packet ratio in scenarios where mobility is high, but has a high overhead due to broadcast messages sent several times. [6] uses the flood process to update the pheromone tables on all nodes, being the packet transmission reach higher than a simple broadcast, but with one overhead greater. [7] presents a robust protocol that provides better quality of service (QoS), but it has a high latency in the route discovery by

being a reactive protocol. HopNet [8] is a highly scalable protocol, but has the disadvantage that when the node number is low, it experiences a greater delay than other protocols because of the continuous movement of peripheral nodes inciting more discovery processes of new routes. However, undoubtedly, the most representative protocol is AntHocNet [3], adaptive and multipath protocol which takes into account the dynamic topology and other characteristics of the MANETs and presents a hybrid operation: reactive because it has agents operating on demand to establish routes to destinations and is proactive because it has other agents which obtain information to discover new alternative routes on prevention by the link failures. A variant of AntHocNet is AntOR, protocol which the present work is based on.

3. AntOR

AntOR is a hybrid ACO protocol which for its properties is adapted to the MANETs. It has the following characteristics which distinguishes it from AntHocNet:

(i) disjoint-link and disjoint-node protocol;

(ii) separation between the pheromone values in the diffusion process;

(iii) use distance metric in route proactive exploration.

Disjoint-Link version (AntOR-DLR) is that in which the links are not shared. In the Disjoint-Node version (AntOR-DNR) are nodes that are not shared. Every disjoint-node is also a disjoint-link, but not vice versa. The two types of routes have the following advantages.

(a) When a node fails, it will only affect a route, but not to the whole network.

(b) The load balancing is better with the disjoint property, because routes are not repeated.

Although it has some disadvantages such as the need for more resources by not sharing links or nodes. In [9], we are seeing a comparison whereby we can see how Link-disjoint improved Node-disjoint. Below a more detailed comparison of these two versions is presented. In the comparison we have used the following two metrics.

(i) *Delivered data packet ratio*: relationship between the number of packets sent and the number of packets delivered successfully.

(ii) *Average end-to-end delay*: measure of accumulative effectiveness of experienced delays by the packets going from source to destination.

Network Simulator NS-3 has been used (specifically version 8) [10]. Simulation parameters are as follows: we have utilized 100 nodes configured according to the Standard IEEE 802.11b, moving in a random scenario dimensions 1000 m $\times$ 1000 m according to the pattern of mobility *Random WayPoint* (RWP). The application of data traffic is *Constant Bit Rate* (CBR) with a packet sending rate of 2048 bps (4 packets of 64 bytes per second). We apply 5 random data session, where mobility is variable from 0 m/s up to a maximum of 10 m/s. Total simulation time is 120 s and pause time at intervals of 30 seconds from 0 s to a maximum of 120 s has been varied. In Figures 1 and 2, you can observe how AntOR-DLR improves AntOR-DNR according to the performance metrics of delivered data packet ratio and average end-to-end delay. More specifically, in Figure 1 we appreciate how the delivery of data packets is better in link-disjoint version than in node version, being significantly higher in simulations where the pause time is scored in 30 and 60 s.

In addition, in Figure 2 we see how the delay is clearly lower, but as it increases the pause time delays are approaching, but fail to match.

These two figures give us information on how the link-disjoint routes have better performance by the failures of link/node. This is due to the fact that the failure of the node-disjoint route is more frequent (as link-disjoint routes serve themselves from independent links that use other nodes).

4. AntOR Disjoint-Link Restrictive (AntOR-RDLR)

This restrictive version, AntOR-RDLR (restrictive disjoint-link route version) covers two characteristics that differentiate it from its predecessor. Firstly, it is the pheromone update process, and on the other hand, the so-called link-Disjoint restrictive property. Thus, in AntOR [2] same route cannot have regular and virtual pheromone simultaneously. In AntOR the updating is in the following way, knowing that the regular pheromone takes precedence over the virtual.

(a) If the node A, which has a route to the destination D, already has regular pheromone, and it reaches virtual pheromone in pheromone diffusion process, then the virtual value is not updated on node A. Therefore, the value of final virtual pheromone is zero, as is shown in

$$\begin{aligned} \text{Regular}_{\text{final}} &= \text{Regular}_{\text{old}} \\ \text{Virtual}_{\text{final}} &= 0. \end{aligned} \tag{1}$$

(b) If the node A, which has aroute to the destination D, already has virtual pheromone, and it gets regular pheromone in the route discovery process, then the value of virtual pheromone is replaced by the value of regular pheromone that arrives. Therefore, the new value of virtual pheromone is 0, as it picks up from

$$\begin{aligned} \text{Regular}_{\text{final}} &= F(\text{Regular}_{\text{new}}, \text{time}) \\ \text{Virtual}_{\text{final}} &= 0. \end{aligned} \tag{2}$$

In the new protocol, AntOR-RDLR, the updating process is as follows.

(a) If the node A, which has a route to the destination D, already has regular pheromone, and it reaches virtual pheromone in pheromone diffusion process, then

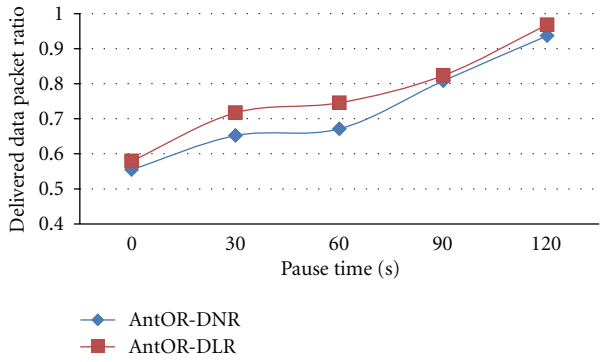


Figure 1: Pause time against delivered data packet ratio.

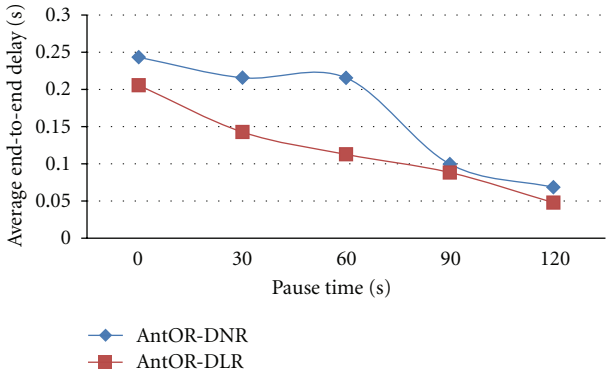


Figure 2: Pause time against average end-to-end delay.

the virtual value is not updated on node A. Therefore, the value of final virtual pheromone is zero, as it is shown in

$$\begin{aligned} \text{Regular}_{\text{final}} &= \text{Regular}_{\text{old}} \\ \text{Virtual}_{\text{final}} &= 0. \end{aligned} \tag{3}$$

(b) If node A has a route to the destination D, it already has virtual pheromone, and it gets regular pheromone in the route discovery process, then the value of regular pheromone replaces the virtual pheromone by the maximum of the value of regular pheromone that arrives and the average between the value of regular pheromone that arrives and the old virtual old, setting the value of virtual pheromone equal to 0, as it picks up from

$$\begin{aligned} \text{Regular}_{\text{last}} &= F(\text{Regular}_{\text{new}}, \text{time}) \\ \text{Regular}_{\text{final}} &= \max(\text{Regular}_{\text{last}}, \text{mean}(\text{Regular}_{\text{new}}, \text{Virtual}_{\text{old}})) \\ \text{Virtual}_{\text{final}} &= 0. \end{aligned} \tag{4}$$

With regard to the restrictive property about link-disjoint routes in AntOR-DLR (disjoint-link route) a same route to a destination cannot share links as shown in next Algorithm 1.

Proactive agents (ants) go by ways which are not link-disjoint.

It is allowed AntOR-RDLR to choose disjoint links for the data retransmission up to a maximum of attempts MAX_HOP_RETRY according to following Algorithm 2.

For example, whether in the proactive retransmission process a disjoint-link route has been selected, in theory, it would not be a candidate to be forwarded, according to the original version of AntOR, but this new version can forward up to a maximum of attempts MAX_HOP_RETRY by the route.

5. AntOR-RDLR versus AntOR-DLR

We then present a comparison of these protocols. In this we have taken into account the following metrics.

(i) *Delivered data packet ratio*: relationship between number of packets sent and the number of packets delivered successfully.

(ii) *Throughput*: volume of work or information flowing through a system. It is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.

(iii) *Overhead in number of bytes*: relationship between the total number of transmitted control bytes and delivered data bytes.

For this comparison the network simulator NS-3 (specifically version 8) [10] has also been used. Simulations parameters are as follows: we have used 100 nodes configured according to the Standard IEEE 802.11b, moving in a random scenario with dimensions of 1000 m × 1000 m according to the mobility pattern *Random WayPoint* (RWP). The application of data traffic is *Constant Bit Rate* (CBR) with a rate of sending packages 2048 bps (4 packets of 64 bytes per second). We apply 5 random data sessions, where mobility is variable from 0 m/s up to a maximum of 10 m/s. Total simulation time is 120 s and pause time has been changed at intervals of 30 seconds from 0 s to a maximum of 120 s. We have done two kinds of experiments. Firstly, an initial experiment in which we wanted to compare the link-disjoint version and its restrictive version. For this comparison we have established MAX_HOP_RETRY at a constant value of 5 attempts. According to the Figures 3 and 4 the restrictive version wins the original version, link-disjoint route, according to metrics of the delivered packet ratio and throughput.

This makes us see that this restrictive version, AntOR-RDLR, behaves more efficiently, providing a better service because fewer packets are lost. This is especially due to AntOR-RDLR has the restrictive property of Link-disjoint routes, already mentioned previously, which makes it possible to create more alternative routes, providing more security by the link/node failures. The second experiment claimed to analyze the evolution of the restrictive version. To perform this comparison MAX_HOP_RETRY from 2 attempts up to a maximum of 10 has been varied, with a pause time of a constant value of 30 s (25% of the total simulation time).

```
While Proactive Process do
  if Link ≠ Session Source then
    Send Control Packet;
  else
end
```

Algorithm 1: Proactive process in AntOR-DLR.

```
HOP_NUM = 0;
While Proactive Process do
  if Link ≠ Session Source then
    Send Control Packet;
  else
    if HOP_NUM≤MAX_HOP_RETRY then
      HOP_NUM = HOP_NUM + 1;
      Send Control Packet;
    end
  end
end
```

Algorithm 2: Proactive process in AntOR-RDLR.

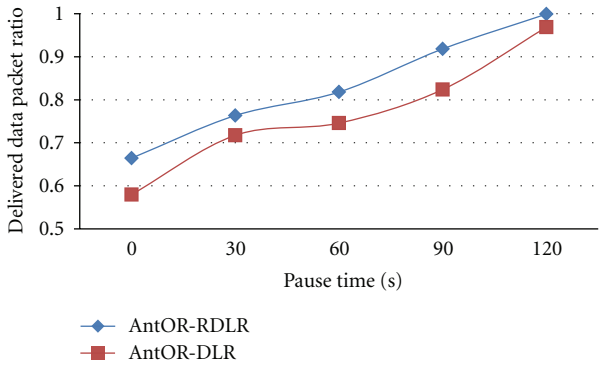


Figure 3: Pause time against delivered data packet ratio.

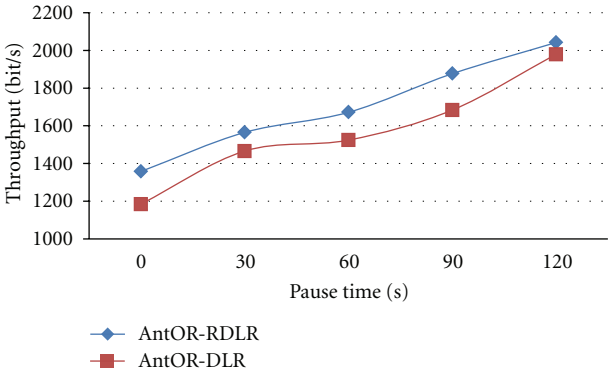


Figure 4: Pause time against throughput.

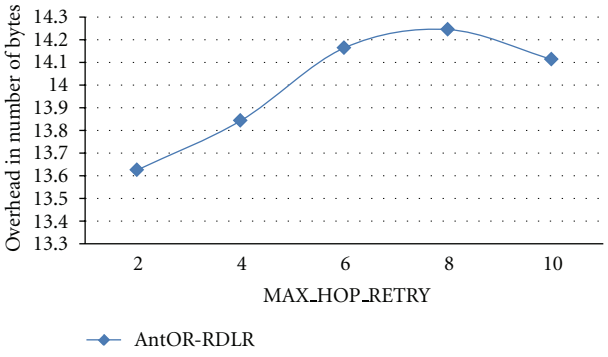


Figure 5: MAX_HOP_RETRY against overhead.

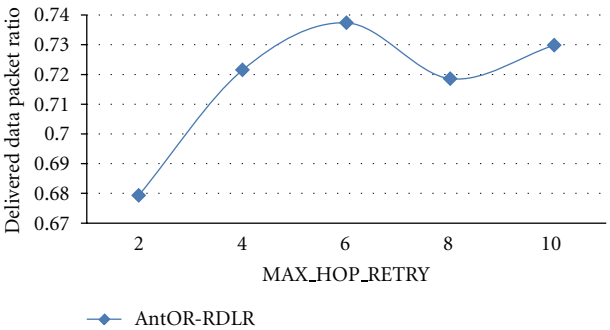


Figure 6: MAX_HOP_RETRY against throughput.

In Figure 5, we observe how the overhead in bytes increases as the number of attempts decreases, but it decreases after 8 attempts. This makes us see that from a given value MAX_HOP_RETRY, we do not improve the performance of the algorithm.

In Figure 6, we have delivered packet ratio. In this graph we can see how the ratio increases according to the number

of attempts until reaching a value of 6. From this value, the ratio shows irregular behaviour.

6. Conclusions

In this work a family of bioinspired routing protocols for mobile ad hoc networks has been presented. The base protocol, called AntOR, has two versions, the so-called link-disjoint (AntOR-DLR) and node-disjoint (AntOR-DNR). A comparison between these versions have been presented observing how Link-disjoint version (AntOR-DLR) improves to node-disjoint version (AntOR-DNR), because the link-disjoint routes have better performance by the link/node failures or, in others words, a node failure occurs more frequently than link failures since link-disjoint routes serves themselves of independent links which use other nodes. Also, a new version of AntOR, which improves the previous ones, has been presented. This new protocol, called AntOR-RDLR, differs from its predecessor, AntOR-DLR in the pheromone updating process and the route discovery mechanism. It has been shown how AntOR-RDLR improves AntOR-DLR in service performance and how to vary the number of attempts MAX_HOP_RETRY is a very important decision in the functioning of the algorithm, because we allow to generate more alternative routes than in AntOR-DLR version.

Acknowledgment

This work was supported by the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID Mediterráneo A1/037528/11.

References

[1] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, 2004.

[2] L. J. García Villalba, D. R. Ruperez Cañas, and A. L. Sandoval Orozco, "Bio-inspired routing protocol for mobile ad hoc networks," *IET Communications*, vol. 4, no. 18, pp. 2187–2195, 2010.

[3] F. Ducatelle, *Adaptive routing in ad hoc wireless multi-hop networks [Ph.D. thesis]*, Universitá della Svizzera Italiana, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, 2007.

[4] O. Hussein and T. Saadawi, "Ant routing algorithm for mobile ad-hoc networks (ARAMA)," in *Proceedings of the 22nd IEEE International Performance, Computing, and Communications Conference*, pp. 281–290, New York, NY, USA, April 2003.

[5] J. S. Baras and H. Mehta, "A probabilistic emergent routing algorithm for mobile ad hoc networks," in *Proceedings of the International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, pp. 3–5, INRIA, Sophia-Antipolis, Paris, France, March 2003.

[6] M. Günes, U. Sorges, and I. Bouazizi, "ARA—the ant-colony based routing algorithm for MANETs," in *Proceedings of the International Workshop on Ad Hoc and Sensor Networks (ICPP '02)*, 2002.

[7] L. Liu and G. Feng, "A novel ant colony based QoS-aware Routing algorithm for MANETs," in *Proceedings of the 1st International Conference on Natural Computation (ICNC '05)*, vol. 3612, pp. 457–466, Springer, Berlin, Germany, August 2005.

[8] J. Wang, E. Osagie, P. Thulasiraman, and R. K. Thulasiram, "HOPNET: a hybrid ant colony optimization routing algorithm for mobile ad hoc network," *Ad Hoc Networks*, vol. 7, no. 4, pp. 690–705, 2009.

[9] D. Rupérez Cañas, A. L. Sandoval Orozco, L. J. García Villalba, and T.-H. Kim, "A comparison study between AntOR-disjoint node routing and AntOR-disjoint link routing for mobile ad hoc networks," in *Proceedings of the FGIT-MulGraB*, vol. 2, pp. 300–304, 2011.

[10] The NS-3 network simulator, http://www.nsnam.org/.

Technique to Neutralize Link Failures for an ACO-Based Routing Algorithm

Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, and Luis Javier García Villalba

Grupo de Análisis, Seguridad y Sistemas (GASS)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática, Despacho 431
Universidad Complutense de Madrid
C/ Profesor José García Santesmases s/n
Ciudad Universitaria, 28040 Madrid, Spain
{delfinrc,asandoval,javiergv}@fdi.ucm.es

Abstract. Ad hoc networks are formed by wireless devices distributed without a predefined infrastructure using a technique called multi-hop communication. A particular case is mobile ad hoc networks, which operate within dynamic environments. This determines the necessity of paying special attention to the routing problem. Traditional techniques are not particulary efficient at making the bioinspired algorithms more relevant. These techniques are based on the analysis of the behavior of some animals, especially in the process of obtaining food. A set of these techniques or algorithms are known as the ACO (Ant Colony Optimization) which is based on the particular behavior of ants. A representative protocol from this kind is AntOR, routing protocol for mobile ad hoc hybrid, multipath and adaptive. In this article a variant of AntOR is proposed which causes the protocol AntOR-UDLR. This approach consists of replacing the link failure notification messages sent in a broadcast manner by unicast messages, which are sent to the predecessor of the node reporting the link failure, until the source of the data session is reached. The simulation results show that AntOR-UDLR improves its predecessor according to all analyzed metrics.

Keywords: bioinspired algorithm, routing protocol, mobile ad hoc networks, ant colony optimization, link failure, unicast.

1 Introduction

Given dynamic topology of mobile ad hoc networks, that is, given the continuous joining and departing of nodes in a mobile ad hoc network, designing of efficient routing protocols is not an easy task because it is not directly applicable for standard routing solutions. There is a group of algorithms called Bioinspired which have their adaptive capabilities as a main characteristic. This proves particularly relevant in this type of environment. Within these algorithms there has been particular reference made in literature to the concept of *Swarm* Intelligence

J. Pavón et al. (Eds.): IBERAMIA 2012, LNAI 7637, pp. 251–260, 2012.

[1], inspired by the social behavior of insects and other animals to solve complex problems. *Swarm* Intelligence is a set of methods to solve difficult optimization problems both static and dynamic problems using cooperative agents, usually called ants. These model a stigmergy behavior, which means the collaboration through a physical medium. Each insect smells the pheromone trail that other ants leave. This seemingly simple behavior solves complex problems. The ant behavior that they carry out of acquiring the food is the principle of the *Ant Colony Optimization* (ACO) algorithms [2]. This algorithm makes reference to the concept of ant as the agent that plays a particular role. It also uses the concept of *forward* ant (that goes from the source node to the destination) and *backward* ant (in the opposite direction).

This article presents the so-called protocol AntOR-UDLR (AntOR - Unicast Disjoint Link Route), variant of AntOR-DLR [3]. The main idea of this algorithm is to replace the link failure notification messages sent in broadcast manner by one-hop and unicast messages sent to the predecessor node of a valid path to a reachable destination. This paper is divided into 5 sections with the first section introducing the concept. In Sect. 2 we discuss the most relevant related work in the routing based on ACO. In Sect. 3 we present AntOR-UDLR explaining the major differences with regard to AntOR-DLR. In the following Sect. 4 we analyze the results of the simulation where AntOR-UDLR, AntOR-DLR and OLSR are compared. Finally we offer conclusions and lines of future work in Sect. 5.

2 Related Work

Many ACO algorithms have been proposed in the literature. These algorithms can be classified, as well as the traditional ones, in proactive, reactive, and hybrid. Proactive protocols frequently need to exchange packets between mobile nodes and to continuously update their routing tables. On the other hand, reactive protocols are that deal of reducing the overhead which produce proactive protocols but they have more latency. As a combination of proactive and reactive part we have hybrid protocols, among it, the following should be noted.

Ant-AODV [4], hybrid routing protocol based on ACO and on the routing protocol AODV, as its name suggests, it tries to take advantage of both. To overcome some of the disadvantages of AODV, as is the overhead generated by the increase of control message, this hybrid technical is utilized, that highlights the node connectivity and decreases the End-to-End delay, as well as the latency of route setup process. Ant-AODV similarly to other protocols such as ADRA [5] was designed without taking into account techniques to help to find the shortest routes and mechanisms to mitigate the congestion problem.

HOPNET [6] is based on a technique in which the ants jump from one zone to another one. The algorithm has characteristics extracted from the ZRP and DSR protocols, being highly scalable, compared with other hybrid protocols. This algorithm consists of proactive route setup in the area of node vicinity, and communication between zones reactively on demand is done when it sends packets from a zone to another. When the number of nodes is small the continuous

movement of peripheral nodes constantly attempts to discover new routes, which causes more delay than in other hybrid routing protocols.

But undoubtedly the most representative is AntHocNet [7]. It constitutes a hybrid, adaptive and multipath protocol that takes into account the dynamic topology and other characteristics of the MANETs, presenting a hybrid mode of operation: it is reactive because it has agents operating in the route setup to destinations and proactive due to other agents collecting information to discover new routes in the prevention against link failure. It is multipath because it establishes different paths to send the information to the destination. Finally, it is adaptive because it suits the traffic and network conditions. In the operation of AntHocNet the following steps or phases can be distinguished:

- Routing information setup: The source node sends reactive agents to discover the first available route to the destination.
- Data routing: Data is sent through the nodes to the destination using the route information and can use a multihop technique, which involves sending data through intermediate nodes. These nodes act as routers.
- Path maintenance and exploration: Information about existing routes is proactively updated and the discovery of new ones is possible.
- Management of link failures: Management failures occur when a node is outside the scope of the network or does not receive control messages which are responsible for informing a node of its closest neighbours (who are one hop), and so on. This phase deals with such failures.

However, it is necessary to improve some aspects as the overhead produced in the route setup phase. This overhead is produced because it does not include techniques to monitor the number of ants that move over the network. Also, the use of disjoint route could improve the efficiency of the algorithm.

Finally, AntOR [3] is a protocol based on AntHocNet but it differs from this in the following characteristics: i) it is a protocol that works in two separate modes: Disjoint-link and Disjoint-node; ii) it takes into account the pheromone separation in the diffusion process; iii) Use of the *distance* metric in path exploration. In such protocol there are two kinds of routes: Disjoint-node and Disjoint-link. The first corresponds to routes in which nodes are not shared and the latter refers to routes in which links are not shared. In AntHocNet a same route can simultaneously have regular and virtual pheromone values. In the proposed protocol a route cannot have both a regular pheromone value and a virtual pheromone simultaneously; this technique improves the efficiency of the algorithm. Finally, it uses the *distance* metric, where AntOR takes into account the number of hops for the routes which have been found to be the best. In this manner, a proactive ant is controlled to ensure it does not go through more nodes than those established by the limit of the number of hops. This hop limit is established according to the best routes (less distance in number of hops) previously calculatedy.

3 AntOR-UDLR

In this article we present a new protocol which is a variant of AntOR-DLR [3]. We chose AntOR-DLR instead of AntOR-DNR because of the comparison done in [8] which showed that it is more beneficial. One of the aspects that the design of this new algorithm pursues is to reduce the overhead in the network. Before specifying it we must differentiate between *unicast* and *broadcast* messages. *Unicast* means that the information from a unique sender to a unique receiver is sent, unlike the broadcast system that sends data to the whole network in an indiscriminate way. *Unicast* mode checks through control messages that the channel is free to transmit. This fact implies more delay to have with respect to broadcast messages, but it has the advantage that it produces fewer collisions, losing fewer messages.

3.1 Specification

The main idea of this approach is to replace the notification messages sent in *broadcast* mode by simple messages sent the precursor of a valid path to a reachable destination. We mean with valid route that route with has pheromone value greater than zero and belongs to the active session of a particular destination. When a node detects the link failure in its neighbour, it communicates such a failure to its predecessor through a *unicast* message, repeating that message to its predecessor until the source node of the data session is reached. This causes the source to launch a *forward* ant in the route setup phase. It may be the case that the node, that detects the link failure, has two or more overlapped data sessions. This causes the failure communication to have to do different predecessors, due to the distinct source data session. Next we explain how to manage the link failures in AntOR-DLR and AntOR-UDLR.

Link Failure Management: In mobile ad hoc networks the link failures can occur by physical changes, such as when a node is switched off or moved, or due to changes that affect the connectivity of wireless communication, such as the increase in the transmission range or a decrease in the utilized transmission power. Since the MANETS are dynamic, these events occur frequently, and the routing algorithms of such a network must be prepared to deal with them efficiently. The first step in the management of link failures is the detection. Once the failure is detected, the next step is the neutralization of the failure. This stage is where AntOR-UDLR differs from AntOR-DLR, significantly improving the performance of the routing algorithm. Then we enter the core concepts of link failures management.

Link Failure Management in AntOR-DLR: Before analyzing the link failure management in this protocol, we should comment AntOR-DLR offers some basic protection components. These components are the route setup process and proactive route maintenance process. The first one allows the source nodes to rebuild the entire route if needed and the second one provides protection in a proactive manner through the creation of new paths, which can serve as backup

for the routing. In AntOR-DLR the link failure is detected whether protocols from the lower layers inform of the transmission failure about control or data packet, or if a node does not receive the corresponding message HELLO from its neighbors. As mentioned, when a failure is detected, we process it to neutralize it. At this phase, AntOR-DLR behaves in the the following way. In AntOR-DLR, the first task that occurs when there is a failure node is that the node detecting the failure removes it from its neighbors table. Then the routing table with the new pheromone information is updated. Finally, it is responsible for neutralizing the failure taking into account the following factors:

a. If there is no route at the source a reactive *forward* ant is sent.
b. If there is no route at an intermediate node and it is dealt with by a data packet that had been forwarding when the failure occurred, a route repair *forward* ant is sent. If there is no reply from the corresponding repair *backward* ant in a certain time period a link failure notification message is sent in broadcast mode, reporting the unreachable destination.
c. When there is a link failure, due to the fact that the corresponding consecutive message HELLO has not been received in a while or because a unicast control message is lost, and if it is dealt with intermediate nodes in the following way is processed: a link failure notification message is created informing about unreachable destinations and this message in broadcast mode is sent.

Link Failure Management in AntOR-UDLR: The algorithm of link failure detection is the same as in AntOR-DLR. The only thing that changes is the way to deal with the corresponding failure. Here is where we comment on the new characteristics.

As AntOR-DLR, the first fact that occurs when there is a node failure in AntOR-UDLR is the node that perceives the failure removes it its neighbors table. Then the routing table with the new pheromone information is updated. Finally, it is processed similarly to AntOR-DLR:

a. If there is no route at source node, a reactive *forward* ant is sent.
b. If there is no route at an intermediate node, and it is dealt with by a data packet that had been forwarding when the failure occurred, a route repair *forward* ant is sent. If there is not reply from the corresponding repair *backward* ant repair in a given time period, a message in *unicast* mode to the precursor of the route is sent informing about the unreachable destination. The node that receives this message updates the routing table and forwards this message to the precursor of the route to the destination. This process is repeated as many times as needed until the source node of the data session is reached.
c. If there is no route at the intermediate node, and it is dealt with by a control packet (a HELLO message is not consecutively received at every certain interval or a unicast control message), no message is not sent. This last option may prevent those routes from repairing correctly, because in this case any operation is not performed. To fix this we create a new functionality: when an intermediate node, which is routing data and does not find a valid route,

i.e., a route with a regular pheromone value greater than zero, sends a *unicast* message to all one-hop neigbours, so that they update their routing tables. It is necessary to send this message to all neighbors, because otherwise we do not have information of the predecessor by not finding a valid route. When one of these neighboring nodes has a valid route to the destination, it forwards the *unicast* message the precursor of the route. This process is repeated as often as needed until the source node is reached.

3.2 Algorithm Design

AntOR-UDLR has a *unicast* message of link notification (ULN), which has a simple structure. It contains two IP addresses: *Session Destination Address* and *Session Source Address*. The first address makes reference to the destination of the data session with a valid route and the second to the source. These two addresses are essential for the functioning of the algorithm. We use the destination address because, when there is a link failure, the node, which detects it, has to indicate the destination in that message, so that the predecessor nodes can process it properly and decide if they must forward such a message depending on whether they detect or not valid route to that destination. The source address is important because it indicates to the node that receives the message ULN, if the source node has been reached or not, by checking if the source address encapsulated in the message is the same than the main address from the node.

```
1  {src, dst} = GetInformation(ULN)
2  if CheckValidRoute(dst) then
3  |  if CURRENT_NODE = src then
4  |  |  Send(RFA)
5  |  else
6  |  |  TTL = TTL - 1
7  |  |  { pre } = GetPrecursor(dst)
8  |  |  ReSend (pre,ULN)
9  |  end
10 end
```

Algorithm 1: Link failure neutralization process

According to Algorithm 1, in line 1 we get the source and destination associated with the data session. This information is extracted from the ULN message. In line 2 we check if there is a valid route (active session and value of regular pheromone greater than zero) to the destination *dst*. In positive case, in line 3 whether the current node (which receives the message ULN) is equivalent to src is checked. If we have reached the source of the data session we perform a new

route setup (line 4), in case contrary we re-forward message ULN (lines 6-8). In 6 we subtract a unit to the value of the field Time to live (TTL). This field in the packet header is included. In line 7 we obtain the precursor *pre*, so that in 8 we forward message ULN to such a precursor.

3.3 Functioning

The following example explains how to treat a link failure at an intermediate node, when data message is transmited and we cannot fix the route (case b of "Link failure management in AntOR-UDLR"). Figure 1 shows a network formed by 5 nodes, where the source and destination node corresponds to the letters A and E respectively. We mark the failed or deactivated node in red which causing the link failure between C and D. Node C notifies its predecessor B, with a simple *unicast* message, that the destination E is unreachable. On receiving the node B this message forwards it to its predecessor A discounting a unit to the TTL value of such a message. Finally when A receives this message and executes a new route setup process because it is the source node.

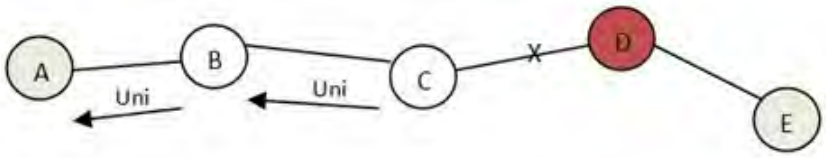
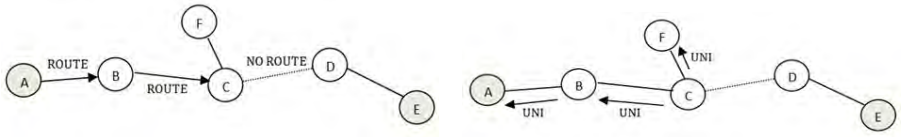


Fig. 1. Example of Link-failure Manage (I)

Figures 2(a) and 2(b) show another scenario in which the new functionality is seen (case c of "Link failure management in AntOR-UDLR" in Sect. 3.1) This scenario (see Fig. 2(a)) consists of 6 nodes where the source and destination of a session of data are represented by the letters A and E respectively. A node forwards the data packet to the reachable destination E through the next hop B.



(a) (b)

Fig. 2. Example of Link-failure Manage (II)

Upon receiving node B the data packet correctly, forwards it to C with destination E. Now node C has to relay it, but cannot find the route to the next hop

D, so that the information cannot be routed to the destination successfully. At this moment, we apply the new functionality (see Fig. 2(b)), by sending a *unicast* message to all neighbours. To be able to send the corresponding message to all neighbours it must find the IP addresses of each one of them in the neighbors table. A *unicast* message is sent by IP address of the found neighbor, rather than a broadcast message; because the sending in unicast mode is more efficient as explained at the beginning of this section. Nodes F and B receive the message sent by C, but do not receive D because it is removed from the neighbors table of C since it originated the failure. Node F processes the message but does not forward it, because it does not belong to the valid route to the destination E. Instead, node B forwards it to node A, since it belongs to this data session. Upon receiving Node A this message, it sends a *forward* ant to proceed with a new route setup.

4 AntOR-UDLR vs. AntOR-DLR

The characteristics of the simulations in Simulator NS-3 were: We used randomly distributed 100 nodes with transmission range of 300 m. The nodes are moved according to the *Random WayPoint* (RWP) pattern with pause time of 2 s. The scenario was rectangular with dimensions 3000 m $\times$ 1000 m. The speed was variable from a minimum of 0 m/s to 10 m/s. It used 10 random data sessions using the application protocol *Constant Bit Rate* (CBR) beginning to send data at random from 0 s to a maximum of 60 s. The sending rate was 512 bit/s, i.e., sending a packet of 64 bytes per second. The maximum simulation time was established to 300 s. It employed a total of 5 runs in the experiment. Fig. 3 and 4 show the Throughput and ratio respectively. Both have similiar behaviour, but using a different scale. In this scenario and with this new algorithm we achieve a ratio of 77% in adverse conditions of speed and dimensions because it is a rectangular scenario which does not help in the packets reception. The square scenarios have a better node distribution and have a more regular and uniform movement. In Fig. 5 we appreciate how the Average End-to-End Delay is lower in AntOR-UDLR than AntOR-DLR at all times. This behavior makes us think that AntOR-UDLR is more stable than AntOR-DLR in presence of a different mobility pattern. In Fig. 6 we ascertain how the overhead in number of bytes is practically the same in AntOR-UDLR than AntOR-DLR. Also we can see that the overhead is a bit bigger in AntOR-UDLR than AntOR-DLR with scenarios highly dinamics (speeds of 8 and 10 m/s) because of the conectivity losing triggers the AntOR-UDLR's funcionality of sending more *unicast* messages, but it is necessary and it is not injured the analysed previous metrics. With regard to OLSR we can appreciate how AntOR-UDLR and AntOR-DLR improve it in all metrics except Average End-to-End Delay. OLSR is a proactive protocol and it has a high overhead and low delay.

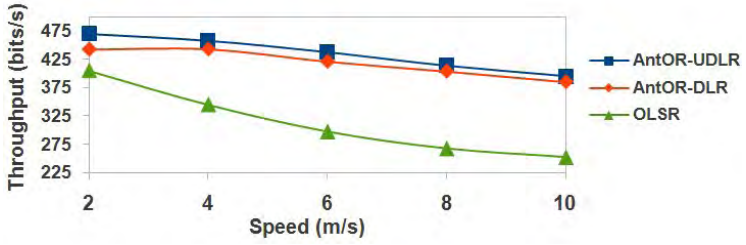


Fig. 3. Throughput

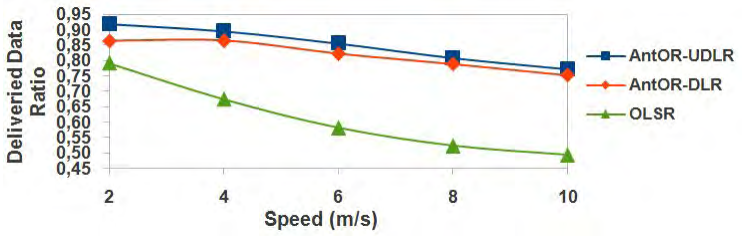


Fig. 4. Delivered Data Ratio

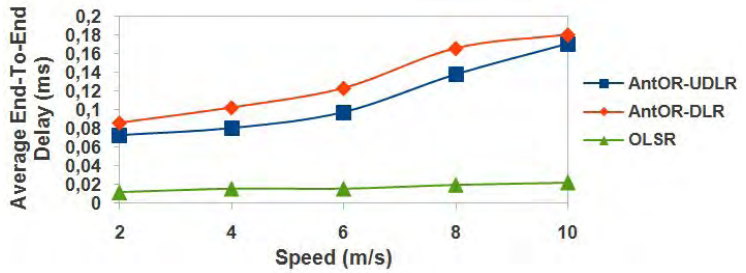


Fig. 5. Average End-To-End Delay

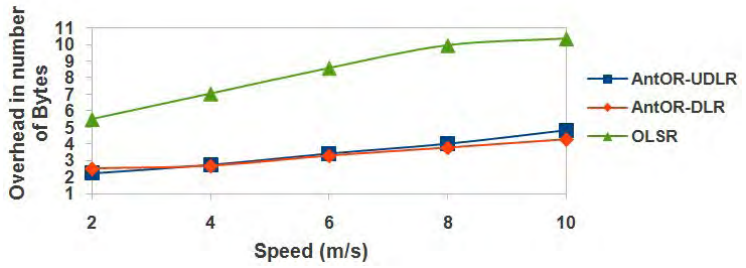


Fig. 6. Overhead in number of Bytes

5 Conclusion and Future Work

In this paper we have presented a new approach, AntOR-UDLR, of routing protocol AntOR-DLR. As bioinspired algorithm it is suited to dynamic environments. The idea of this variant is to replace, in the process of link failure, the failure notification messages sent in *broadcast* mode by *unicast* message, sent to the predecessor to the source, but associated to the destination of a valid route. Also, using *unicast* messages we lose fewer messages, because the medium is checked before transmitting, i.e., if the medium we want to send is free; this fact does not happen when it is sent through *broadcast* mode. With this new protocol we have aimed at achieving the two proposed objectives: to reduce network traffic and to prevent the transmitted information arriving to nodes unnecessarily, i.e., they do not need to process it. The results show that AntOR-UDLR improves AntOR-DLR and OLSR. As future work we aim at replacing local route repair messages by unicast message, as well as evaluating more metrics as could be the Jitter.

Acknowledgments. This work was supported by the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) through the Project Avanza Competitividad I+D+I TSI-020100-2011-165 and the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID MEDITERRÁNEO A1/037528/11.

References

1. Kennedy, J.: Swarm Intelligence. Morgan Kaufmann Publishers (2001)
2. Dorigo, M.: Optimization, Learning and Natural Algorithms. PhD thesis. Politecnico di Milano, Italy (1992)
3. García, L.J., Rupérez, D., Sandoval, A.L.: Bioinspired Routing Protocol for Mobile Ad Hoc Networks. IET Communications 4(18), 2187–2195 (2010)
4. Marwaha, S., Tham, C.K., Srinavasan, D.: Mobile Agents Based Routing Protocol for Mobile Ad Hoc Networks. In: IEEE Global Telecommunications Conference (GLOBECOM 2002), vol. 1, pp. 163–167. IEEE (2002)
5. Zheng, X., Guo, W., Renting Liu, R.: An Ant-Based Distributed Routing Algorithm for Ad-Hoc Networks. In: International Conference on Communications, Circuits and Systems (ICCCAS 2004), vol. 1, pp. 412–417. IEEE (2004)
6. Wang, J., Osagie, E., Thulasiraman, P., Thulasiram, R.K.: HOPNET – A Hybrid Ant Colony Optimization Routing Algorithm for Mobile Ad Hoc Network. Ad Hoc Networks 7(4), 690–705 (2009)
7. Ducatelle, F.: Adaptive Routing in Ad Hoc Wireless Multi-Hop Networks. PhD thesis, Università della Svizzera Italiana, Istituto Dalle Molle di Studi sull Intelligenza Artificiale (2007)
8. Rupérez, D., Sandoval, A.L., García, L.J., Kim, T.H.: A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks. In: Tai-hoon, K., et al. (eds.) MulGraB 2011, Part II. CCIS, vol. 263, pp. 300–304. Springer, Heidelberg (2011)

Parallel approach of a bioinspired routing protocol for MANETs

Luis Javier García Villalba*, Delfín Rupérez Cañas and Ana Lucila Sandoval Orozco

Grupo de Análisis, Seguridad y Sistemas (GASS)
Departamento de Ingenierla del Software e Inteligencia Artificial (DISIA)
Facultad de Informática, Despacho 431
Universidad Complutense de Madrid (UCM)
Calle Profesor José Garcia Santesmases s/n
Ciudad Universitaria, 28040 Madrid, Spain
E-mail: javiergv@fdi.ucm.es,
E-mail: delfinrc@fdi.ucm.es,
E-mail: asandoval@fdi.ucm.es
*Corresponding author

Abstract: Designing routing protocols for Mobile Ad Hoc Networks (MANETs) is a complex task because of its dynamic topology. A kind of routing protocols that suits the particularity of MANETs is so-called bio-inspired. Among these, focused on Ant Colony Optimisation (ACO), which studies the behaviour of ants in their search for food, are especially relevant. One of these algorithms is AntOR, which relying on swarm intelligence, efficiently solves routing in MANETs. In this paper we show a parallelised version of AntOR, the so-called P-AntOR, that using programming multiprocessor architectures based on shared memory protocol, allows to run tasks in parallel using threads, being applicable this parallelisation in the route discovery phase, route local repair process and link failure notification. The simulation results indicate that P-AntOR performs better than its predecessor, with emphasis on the metric of average End-To-End delay, *jitter* and packet delivery ratio.

Keywords: Parallel protocol; bioinspired routing; MANETs; swarm intelligence; thread; ant colony optimization; shared memory.

Reference to this paper should be made as follows: García Villalba, L.J., Rupérez Cañas, D. and Sandoval Orozco, A.L. (2013) 'Parallel approach of a bioinspired routing protocol for MANETs', *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 12, No. 3, pp.141–146.

Biographical notes: Luis Javier Garcia Villalba received a Telecommunication Engineering degree from the Universidad de Málaga (Spain) in 1993 and holds an MSc. in Computer Networks (1996) and a PhD in Computer Science (1999), both from the Universidad Politecnica de Madrid (Spain). He is a Visiting Scholar at COSIC (Computer Security and Industrial Cryptography, Department of Electrical Engineering, Faculty of Engineering, Katholieke Universiteit Leuven, Belgium) in 2000 and Visiting Scientist at IBM Research Division (IBM Almaden Research Center, San Jose, CA, USA) in 2001 and 2002. He is currently Associate Professor of the Department of Software Engineering and Artificial Intelligence at the Universidad Complutense de Madrid (UCM) and Head of Complutense Research Group GASS (Group of Analysis, Security and Systems), which is located in the School of Computer Science at the UCM Campus. His professional experience includes research projects with Hitachi, IBM, Nokia and Safelayer Secure Communications. His main research interests are: a) cryptography, coding, information security and its applications and b) Mobile Ad Hoc Networks (MANETs), Wireless Sensors Networks (WSNs) and Next Generation Networks (NGNs).

Delfin Rupérez Cañas received a Computer Science Engineering degree from the Universidad Complutense de Madrid (UCM, Spain) in 2007. He holds an MSc in Research in Computer Science from the Universidad Complutense de Madrid (Spain) in 2009. He is a Visiting Scholar at the School of Computing (Portsmouth University, UK) in 2010. He is currently a PhD Student at the Universidad Complutense de Madrid (Spain) and a Research Assistant at Complutense Research Group GASS. His main research interests are: a) Mobile Ad Hoc Networks (MANETs), b) Next Generation Networks (NGNs) and c) privacy.

Ana Lucila Sandoval Orozco received a Computer Science Engineering degree from the Universidad Autónoma del Caribe (Colombia) in 2001. She holds a Specialisation Course in Computer Networks (2006) from the Universidad del Norte (Colombia) and holds an MSc in Research in Computer Science from the Universidad Complutense de Madrid (Spain) in 2009. She is currently

a PhD Student at the Universidad Complutense de Madrid (Spain) and a Research Assistant at Complutense Research Group GASS. Her main research interests are: a) Mobile Ad Hoc Networks (MANETs) and Wireless Sensors Networks (WSNs), b) coding theory and c) information security and its applications.

1 Introduction

In the field of wireless communications, ad hoc networks provide flexibility and autonomy for their self-organisation capacity. This requires specific algorithms and routing protocols to successfully solve the communication process between mobile nodes (Juang and Liu, 2002; Huang et al., 2005; Manoharan et al., 2008; Lin et al., 2010; Zahary and Ayesh, 2010; Biradar and Manvi, 2011; Varaprasad, 2011).

The behaviour of the routing protocols for MANETs is often irregular as they have to select different intermediate nodes in charge of forwarding the information. Since the mobility of the nodes is unpredictable, there are frequent changes in the internal process of data relay. Similarly, information exchange among processors shows a frequent load balancing and communication unpredictable. A variant of the routing protocols for MANETs are the so-called bio-inspired, which are those that are inspired by the behaviour of some animals as suitable to solve routing problematic in such networks. Their are development has been linked to the evolution of the parallel computer architectures lately. Within the bio-inspired algorithms, the behaviour of ants is particularly suitable for modelling of MANETs. These algorithms of constructive meta-heuristics search based on this collective, belonging to the technique called ACO (Dorigo and Stutzle, 2004). This technique uses swarm intelligence (Bonabeau et al., 1999), which treats about the social behaviour of insects and other animals to solve problems.

There are several routing algorithms for MANETs, which are a special application of ACO. A bio-inspired routing protocol that uses this technique is AntOR (Garcia Villalba et al., 2010), a variant of AntHocNet (Di Caro et al., 2005; Ducatelle, 2007), adaptive and multipath by using several routes to send data from source to destination in a same data session. To improve the performance of this protocol, we show the specification and implementation of a parallel approach in this paper. The parallelisation of this sequential algorithm was performed in the following phases: in the route discovery process, in the route local repair process and in the actualisation mechanism to link failures.

AntOR is an algorithm that uses node-disjoint and link-disjoint routes. We have taken into account the AnOR-DNR (node-disjoint version) for the specification and implementation of this parallel approach owing to the parallelisation properties of the version in question.

This paper is organised into 6 sections; the first one is this introduction. In Section 2, we present related works, where we expose some of the bio-inspired techniques for parallel protocol based on the behaviour of ants. In section 3, we briefly show the AntOR routing protocol, pointing out the differences with the parallel version. In section 4, we specify the used parallelisation technique. The most significant results of the simulation are discussed in section 5. Finally, section 6 contains the conclusions and possible future work.

2 Related works

ACO is a technique easily parallelisable by its distributed nature. One of the first parallelisation techniques for ACO was introduced by Bullnheimer et al. (1998). This method is particularly suitable to resolve difficult combinatorial optimisation problems based on the technique used by ants. This meta-heuristic method is a set of artificial agents that cooperate together and with a set of rules that determinate the generation of local and global information and its update, with the aim of finding the best solutions. This method has limitations in its development when are analysed issues such as: the number of local iterations, how the allocation rules of tasks to the processors should be, the static/dynamic approaches and so on are analysed.

Stutzle (1998) applies a master/slave approximation to parallel the different searching techniques from ACO solutions with the characteristic that they do not interact. Stuutzle employs a simple strategy to execute the independent and parallel sessions of an algorithm. The empirical tests are performed using *MAX-MIN* Ant System to Travelling Salesman Problem, showing it to be very efficient. In this case, the parallelisation strategy has the drawback that it depends on as the problem itself as the available hardware.

Michel and Middendorf (1998) use ACO to solve *Shortest Common supersequence* (SCS) problem, which has applications in production system planning, mechanical engineering and molecular biology. They use the 'island model' with several populations of ants, suggesting the following method: separate the ant colonies that exchange information according to the trail to follow, but instead of using a graph (representing typical ACO), the authors directly used a representation of the problem through a 'string' and they assign a pheromone value to each character of this string. Their results show this algorithm as a better heuristic and it is compared favourably with a genetic algorithm, but it has the disadvantage that you lose the functionality provided by the use of graphs.

Delise et al. (2001) implement a new ACO parallelisation system for industrial scheduling problems, testing it in a shared memory processor with OpenMP. The algorithm, as sequential optimisation, solves particular problems showing its utility, but requires a considerable computational time and a lot of resources. However, the structure of the algorithm makes it possible to adapt for parallelisation, dramatically improving the results. It is also noted that the results are better when the algorithm execution time is increased.

Randall, (2002) analyses different strategies of parallel decomposition, which specifically apply to the *Travelling Salesman Problem* (TSP). These strategies are only a guide for the parallelisation of the ACO meta-heuristics, so it cannot be considered a formal approach and generic. The results show an acceptable *Speedup*, given that in large problems is better efficiency achieved, but has the disadvantage in that it requires a great amount of information, thus not being scalable.

3 AntOR: bioinspired routing protocol for MANETs

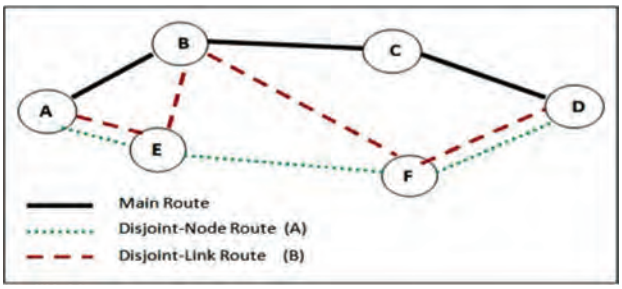
AntOR (Garcia Villalba et al., 2010) is based on the protocol AntHocNet, specifically takes Ducatelle's algorithm as its starting point (Ducatelle, 2007) and has the following differences with respect to this:

a) It is a protocol with the property of link/node disjoint as shown in Figure 1, which provides a better distribution of packet traffic;

b) It has the characteristic that separates the pheromone values in the diffusion process. Thus, a same route cannot have both a regular pheromone value and a virtual pheromone simultaneously;

c) It uses the metric distance in the path exploration. This technique significantly reduces the protocol overhead.

Figure 1 Scenario of node/link-disjoint route (see online version for colours)



AntOR is itself efficient as a sequential algorithm, but it does not take advantage of new features of existing equipment, as the number of cores, the largest amount of memory and so on. For this reason, we decided to make a variant of AntOR that uses multiple processors through threads. We parallelised the node-disjoint version for two reasons:

1 each node-disjoint route includes link-disjoint route and but not vice versa,

2 tolerance to failures is more restrictive.

4 P-AntOR: parallel approach of AntOR

To understand how P-AntOR works, it is necessary to employ three concepts:

1 Process: Program running. The processes are managed by the Operating System.

2 Thread: The basic unit of execution. Any program that executes at least has a thread.

3 POSIX Thread: Standard based in thread API for C/C++.

We use POSIX Thread because it allows a new concurrent process flow to expand. This is the most efficient multi-core systems, where the flow of processes can be scheduled to run on another processor, thus gaining speed through parallel or distributed processing. Programming with threads carries less overhead than expanding a new process, because the system does not initialise a new environment and virtual memory space for that process.

Parallel programming technologies, such as MPI and PVM, are used in a distributed computing environment, while the threads are limited to a single computer system. All threads within a process share the same address space. For the implementation of this routing algorithm to be faster, we use the POSIX Thread library.

Then, we specify a large-grained parallelisation of AntOR-DNR routing algorithm (node-disjoint version). This parallel technique launches a thread for each neighbour that is in the neighbour table of the node that starts one of the following phases:

1 Routing Information Setup: Figure 2 shows a flow chart representing the parallelism in the route discovery process.

2 When a data session is active, the source node is ready to send data to the destination node and the route discovery process is activated. This process is parallelised using threads, so that it launches an ant (agent) reactive through an independent thread to the one-hop neighbours, with the number of utilised threads being proportional to the neighbour number of node, initiating the route discovery. When an intermediate node receives this, ant repeats the process. But, whether it consists in a destination node, this node sends its corresponding *Reactive Backward Ant* (RBA).

3 Local Route Repair: Similar to route discovery, unless it is done locally, as shown in Figure 3

4 Link Failure Notification: The process of link failure notification (see Figure 4) is to update the routing table to the link failures. It is a very important stage and this action must be taken promptly. The nodes send ants through independent threads until an intermediate node has any alternative route to the destination after updating the routing table.

Figure 2 Scheme to parallelise route discovery process (see online version for colours)

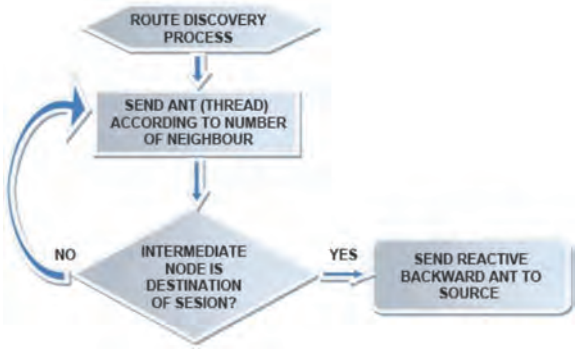


Figure 3 Scheme to parallelise route local repair process (see online version for colours)

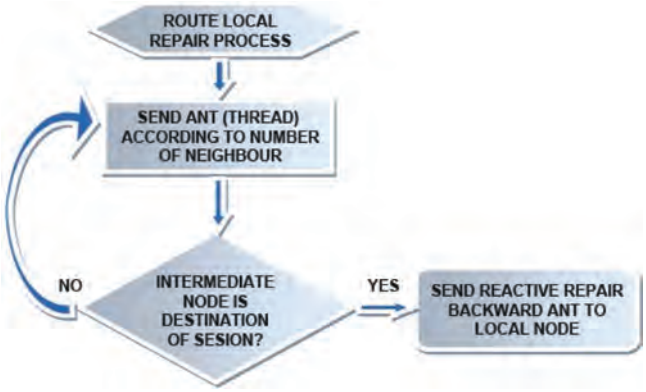
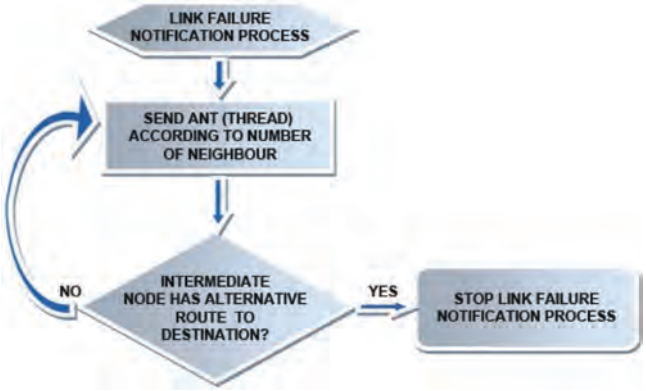
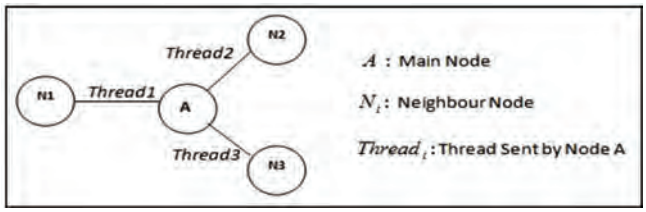


Figure 4 Scheme to parallelise link failure notification process (see online version for colours)



For example: We have a node A that wants to start the route discovery process. In this process, the node consults its neighbour table $N = \{N_1, N_2, N_3\}$, the candidates to send a reactive ant through an independent thread. In this example, 3 threads are sent as shown in Figure 5.

Figure 5 Illustrative example of parallel process



5 Simulation and results

For testing, we have utilised a 4-core processor with 4 GB of RAM using a shared memory system based on POSIX threads standard.

5.1 Analysed performance metrics

The following are the analysed performance metrics to assess our algorithm:

a) Delivered data packet ratio: Relationship between number of packets sent and the number of packets delivered successfully.

b) Average end-to-end delay: Performance metric that measures the accumulative effectiveness of experienced delays by packets going from source to destination.

c) Jitter: Performance metric that measures the delay variation between consecutive packets received. It was considered jitter taking into account RFC 1889 (Schulzrinne et al., 1996).

5.2 Results

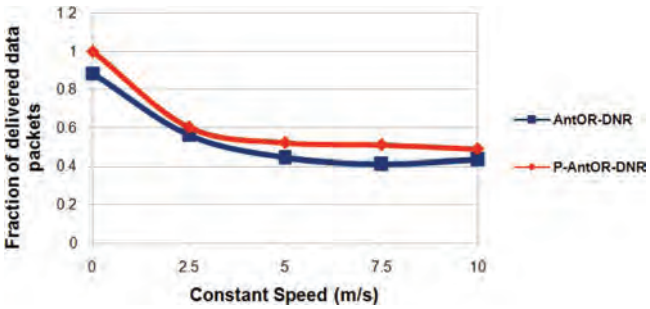
We have carried out several tests with the network simulator NS-3. In this section, we compare P-AntOR with AntOR-DNR (node-disjoint version) to evaluate the performance when using multiple processors.

For the simulation, we use 100 nodes configured according to IEEE 802.11b in an area of 1200 m × 1200 m, in which all nodes are distributed randomly. In the simulation using the traffic generator *Constant Bit Rate* (CBR), are sent 4 packets of 64 bytes of data every second. The total simulation time is set to 120 s. Moreover, we have used 3 tests for each of the performance metrics. We considered two kinds of experiments that use the Mobility Model *Random Wait Point* (RWP). In the first one, we vary the node speed and the second one pause time.

5.2.1 Experiment A: varying the node speed

In this test, the speed varies from 0 to a maximum of 10 m/s. With a pause time of 30 s (25% of the total simulation time). As shown in Figure 6, the ratio in P-AntOR is slightly higher in both a static (0 m/s) and highly dynamic (10 m/s).

Figure 6 Delivered data packet ratio vs. Constant speed (see online version for colours)



The performance metric of average end-to-end delay and jitter (see Figures 7 and 8, respectively) are also better for P-AntOR. For the average end-to-end delay, the improvement is seen at all times, regardless of the speed of the nodes. On the other hand, for the jitter, the results are better from a speed of 7.5 m/s. These results demonstrate that P-AntOR is faster.

5.2.2 Experiment B: varying pause time

In this experiment, we have varied the pause time of nodes from a low of 0 s and a maximum of 120 s. The

Figure 7 Average end-to-end delay versus Constant speed (see online version for colours)

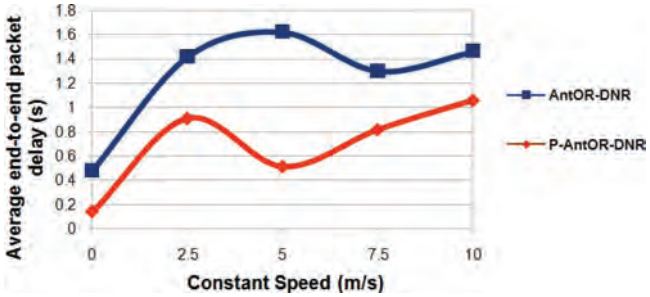
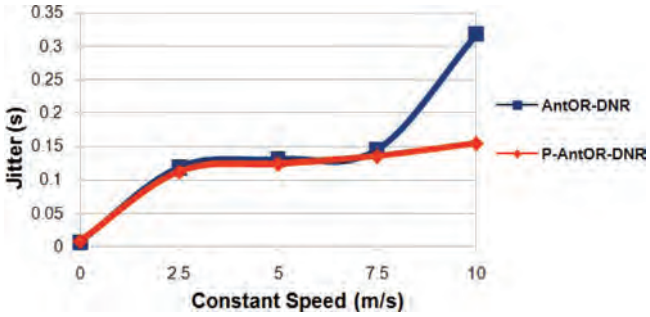


Figure 8 Jitter versus Constant speed (see online version for colours)



increase is pause time has two effects on the general properties of the scenarios that are relevant for routing. The first of these is the decrease in the mobility of the nodes: If the nodes are high pause time, logically they are less mobile on scenario and the network is less dynamic. Consequently, the scenario is less complicated for routing algorithm processing. The second effect is related to the distribution of the nodes in the scenario area when using the RWP mobility model. We have seen that under this model there is a tendency to have a higher density of nodes in the centre area of the network than at the ends, especially when the pause time is low.

Figure 9 shows that P-AntOR has a delivered data packet ratio higher than the sequential version AntOR-DNR. Moreover, the graph of P-AntOR is fairly uniform according to pause time, which does not occur with AntOR-DNR. The metrics in Figures 10 and 11 show how the end-to-end delay and jitter are also lower in P-AntOR.

Figure 9 Delivered data packet ratio vs. Pause time (see online version for colours)

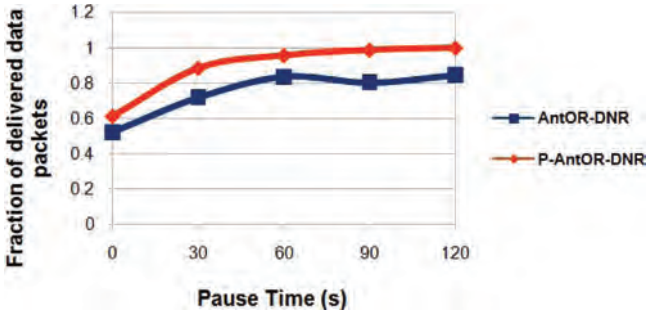


Figure 10 Average end-to-end delay vs. Pause time (see online version for colours)

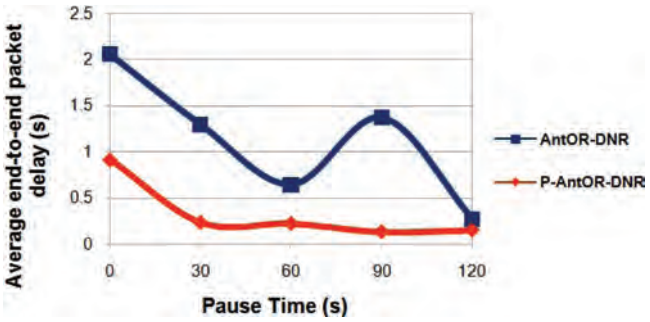
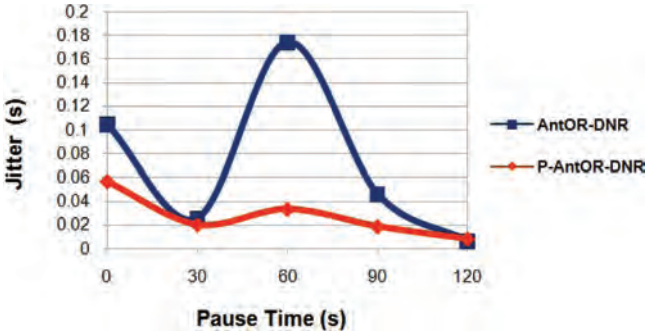


Figure 11 Jitter vs. Pause time (see online version for colours)



6 Conclusions and future work

In this work, a parallel approach of AntOR, called P-AntOR, has been presented. The used parallel technique is a large-grained approach, in which a multi*core* machine in a shared memory system has been used. The essence of this parallel approach is to replace the *broadcast* messages by messages that are sent specifically to one-hop neighbours using threads. The simulation results (it improves analyzed metrics of delivered data packet ratio, average end-to-end delay and jitter) shows, that P-AntOR is more efficient than AntOR-DNR.

As a next step we propose its implementation in actual machines to see the comparison.

For the future, there are several lines of work, such as the use of machines with greater capabilities (number of cores, frequency of cores, more memory and so on) to allow the proposed technique to improve. We can also compare our approach with the parallelism of link-disjoint routes, so that the difference could be seen.

Finally, a parallelisation hybrid system such as that reported by Thakur and Gropp (2009) can be utilised, which consists of a test to evaluate the communication performance *Message Passing Interface* (MPI) multithreading. In this approach, hybrid programming models can be used, combining MPI across nodes and multithreading within a node, because many MPI implementations are beginning to support multithreaded MPI communication. With this technique, better results are probably achieved by more efficient interaction of the nodes.

Acknowledgements

This work was supported by the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) through the Projects Avanza Competitividad I + D + I TSI-020100-2011-165 and TSI-020100-2010-482 and the Ministerio de Ciencia e Innovaciaon (MICINN, Spain) through the Project TEC2010-18894/TCM.

References

Biradar, R.C. and Manvi, S.S. (2011) 'Agent-driven backbone ring-based reliable multicast routing in mobile ad hoc networks', *IET Communications*, Vol. 5, No. 2, pp.172–189.

Bonabeau, E., Dorigo, M. and Theraulaz, G. (1999) 'Swarm intelligence: from natural to artificial systems', *Oxford University Press.* New York, NY, USA.

Bullnheimer, B., Kostis, G. and Strauss, C. (1998) 'Parallelization strategies for the ant systems', *High Performance Algorithms and Software in Non-linear Optimization*, Vol. 24, pp.87–100.

Delisle, P.P., Krakecki, M., Gravel, M. and Gagne, C. (2001) 'Parallel implementation of an ant colony optimization metaheuristic with openmp', *In International Conference of Parallel Architectures and Complication Techniques*, pp.8–12.

Di Caro, G., Ducatelle, F. and Gambardella, L.M. (2005) 'AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks', *European Transactions on Telecommunications*, Special Issue on Self-organization in Mobile Networking, Vol. 16, No. 5, pp.433–455.

Dorigo, M. and Stützle, T. (2004) 'Ant colony optimization', *The MIT Press.* Bradford Company Scituate, MA, USA.

Ducatelle, F. (2007) 'Adaptive routing in ad hoc wireless multi-hop networks', *Ph.D. thesis Universitá della Svizzera Italiana, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale.*

García Villalba, L.J., Rupérez Cañas, D. and Sandoval Orozco, A.L. (2010) 'Bioinspired routing protocol for mobile ad hoc networks', *IET Communications*, Vol. 4, No. 18, pp.2187–2195.

Huang, C., Guo, M. and Chang, R. (2005) 'A weight-based clustering multicast routing protocol for mobile ad hoc networks', *Int. J. of Internet Protocol Technology*, Vol. 1, No. 1, pp.10–18.

Juang, T. and Liu, M.C. (2002) 'An efficient asynchronous recovery algorithm in wireless mobile ad hoc networks', *Journal of Internet Technology*, Vol. 3, No. 2, pp.147–155.

Lin, T., Chao, H. and Woungang, I. (2010) 'An enhanced mpr-based solution for flooding of broadcast messages in OLSR wireless ad hoc networks', *Mobile Information Systems*, Vol. 6, No. 3, pp.249–257.

Manoharan, R., Thambidurai, P. and Pandian, S.L. (2008) 'Energy efficient robust on-demand multicast routing protocol for MANETs', *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 3, No. 2, pp.90–98.

Michel, R. and Middendorf, M. (1998) 'An island based ant system with lookahead for the shortest common subsequence problem', *in Proceedings of the Fifth International Conference on Parallel Problems Solving from Nature (PPSN 1998)*, Vol. 1498, pp.692–708.

Randall, M. (2002) 'A parallel implementation of ant colony optimization', *Parallel and Distributed Computing,* Vol. 62, pp. 1421–1432.

Schulzrinne, H., Casner, S. and Frederick, R. (1996) 'RTP: a transport protocol for real-time applications', *Internet Engineering Task Force* RFC 1889.

Stützle, T. (1998) 'Parallelization strategies for ant colony optimization', *in Proceedings of Parallel Problem Solving from Nature (PPSN 1998)*, Vol. 1498, pp.722–731

Thakur, R. and Gropp, W. (2009) 'Test suite for evaluating performance of multithreaded MPI communication', *Parallel Computing*, Vol. 35, No. 12, pp.608–617.

Varaprasad, G. (2011) 'Network connectivity based power-aware routing algorithm for mobile ad hoc networks', *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 7, No. 2, pp.71–76.

Zahary, A. and Ayesh, A. (2010) 'An analytical review for multipath routing in mobile ad hoc networks', *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 5, No. 2, pp.69–85.

Computing
DOI 10.1007/s00607-013-0310-8

Adaptive routing protocol for mobile ad hoc networks

Delfín Rupérez Cañas · Luis Javier García Villalba · Ana Lucila Sandoval Orozco · Tai-Hoon Kim

Received: 31 January 2013 / Accepted: 10 February 2013

Abstract Artificia immune systems (AIS) are used for solving complex optimization problems and can be applied to the detection of misbehaviors, such as a fault tolerant. We present novel techniques for the routing optimization from the perspective of the artificia immunology theory. We discussed the bioinspired protocol AntOR and analyze its new enhancements. This ACO protocol based on swarm intelligence takes into account the behavior of the ants at the time of obtaining the food. In the simulation results we compare it with the reactive protocol AODV observing how our proposal improves it according to Jitter, the delivered data packet ratio, throughput and overhead in number of packets metrics.

Keywords Ant colony optimization · Artificia immune system · Bioinspired protocol · Mobile ad hoc networks · Routing

Mathematics Subject Classification 68M12 Network protocols

D. Rupérez Cañas · L. J. García Villalba (✉) · A. L. Sandoval Orozco
Group of Analysis, Security and Systems (GASS), Department of Software Engineering and Artificia Intelligence (DISIA), School of Computer Science, Offic 431, Universidad Complutense de Madrid (UCM), Calle Profesor José García Santesmases s/n Ciudad Universitaria, 28040 Madrid, Spain
e-mail: javiergv@fdi.ucm.es

D. Rupérez Cañas
e-mail: delfinrc@fdi.ucm.es

A. L. Sandoval Orozco
e-mail: asandoval@fdi.ucm.es

T.-H. Kim
Department of Convergence Security, Sungshin Women's University,
249-1, Dongseon-dong 3-ga, Seoul 136-742, Korea
e-mail: taihoonn@daum.net

Published online: 16 March 2013

1 Introduction

Optimization problems can be solved by artificia immune systems. These problems we face with these kinds of problems daily: the efficien y improvement of the resources of the devices, fin the shortest path between two points, distribute the resources in the system uniformly.

One of the optimization algorithms based on the colony of ants [1] and that relies on the swarm intelligence [2], has been frequently cited in the literature. It is inspired by the behavior of ants at the time of obtaining the food and in many areas is applied.

ACO algorithms are composed by agents that work without the need of a centralized control structure, in such a way that the interactions local of each an agent and its neighbors allow between them to communication in an autonomous way. These algorithms can be used to resolve routing problems, being suitable for highly dynamic environments. We review the protocol AntOR [3,4], which is based on ant colony optimization (ACO) [1] and it is applied to solve the routing problem in mobile ad hoc networks (MANETs) [5]. This type of networks are formed by wireless devices/nodes which operate in a distributed manner where all these nodes are in the same level, working as clients or servers interchangeably. Also we present improvements to AntOR, in its disjoint-link version and we show its relationship with the artificia immune systems. We structure the rest of article as follows: Sect. 2 includes some relevant works. In Sect. 3 we review briefl our algorithm AntOR presented in the literature and then we explain our techniques applied directly to AntOR and we explain the proposal as a view point of immunology. In Sect. 4 the simulation results in a dynamic environment are exposed comparing them with the standard protocol AODV. Finally, Sect. 5 presents conclusions.

2 Related works

The ACO-based routing protocol can be classified as well as the traditional protocol, in proactive, reactive and hybrid.

Proactive protocols frequently need to exchange packets between mobile nodes and continuously to update their routing tables. It leads a lot of overhead.

Thus, PERA [6] is a proactive protocol in which the route setup is done by two kinds of ants: forward and backward. These agents create and adjust the probability distribution at each node with regards to its neighbors. The node has the probability that each of its neighbors can receive and forward the data packet.

Each forward ant contains the IP addresses from the source and destination nodes, a sequence number, a hop counter fiel and a stack that grows dynamically. The stack contains information about nodes visited by forward and associated times.

A forward ant is created when a node does not have a record of a route to a destination, where the node puts its own IP address onto the stack of such an ant, as well as the time in which the ant is created. From this moment the node periodically saves the forward ants sent to destinations when the route is required. When this forward ant reaches the destination, the destination node creates a new agent, a backward ant. This one uses the information contained in the forward ant in the inverse path, to update the probability distribution in each node and to reflec the current state of the network.

The agent packets, the ants, are of varying size, because it contains different information as the stack that grows dynamically, so it is necessary to allocate memory. Thus, it may cause a high overload in the number of bytes, particularly in highly dynamic environments.

ARAMA [7] is a proactive protocol where the forward ants not only take into account the factor of hop counter (as most of the protocols), but they also valued the link-local heuristic through the route, such as the energy of the node battery and the queue delay. ARAMA define a value called degree. This value is calculated for each backward ant based on information from the path stored in the forward ant. At each node the backward ant updates the amount of pheromone of the routing table from the node using the value degree. The protocol uses the same degree to update the pheromone value of all links. The authors claim that the overhead of the route setup and maintenance is reduced through the control of the forward ant number. However they did not clarify how to control the ant generation in a dynamic environment. Likewise, to note that, although ARAMA optimizes the hop number and the distribution of energy use, it is possible to occur unbalanced load distributions and network bottlenecks.

On the other hand, reactive protocols are that deal of reducing the overhead which produce proactive protocols. To this end, they propose that the nodes only calculate the route to a destination on demand when this node has to begin a packet exchange with the destination. The route setup is normally done by request messages which are floode through the network.

These protocols involve a high latency caused by the route setup.

Among reactive protocols we highlight the following:

ARA [8] is a reactive protocol in which the routing table entries contain pheromone values that facilitate the neighbor choosing. To get a destination it is necessary to choose a neighbor that serves us as link and in this way successively until arriving at the destination. In the routing table pheromone values are degraded over time and the nodes pass to sleep mode if they have reached a threshold which is too low.

In ARA the route setup is done by two mobile agents called forward ants and backward ants. During route setup, forward and backward ants have a unique sequence number, to avoid duplicate packets, being expanded by source and destination nodes through flooding Forward ants and backward ants update pheromone tables over all nodes by flooding The floodin has more packet transmission range, because the packets by floo are transmitted to all nodes in the network via multihops, while the broadcast is transmitted to the one-hop neighbors. The problem of the floodin is that it carries a high overhead. Once route setup has been carried out for a certain destination, the sender node does not generate a new mobile agent toward the destination anymore, but the route maintenance is carried out by data packets.

ADRA [9] uses ants that move through the network between pairs of nodes chosen randomly. These ants when moved deposit pheromone based on several parameters: distance in hops from its origin node, the link quality, the congestion found in its journey, the current pheromone of the node and the speed with which the nodes are moved. Of course, the same node, by the pheromone evaporation, changes its value according to the link age. An ant selects its path at each intermediate node according to the pheromone that the ant has distributed and to speed up the path selection parameters with different values, which update the probability in the routing table is given.

ADRA shows good performance in terms of delivered packet ratio, overhead in control messages and packet end-to-end delay, exhibiting many attractive features of distributed control. However, it has the disadvantage that it does not solve the congestion problem produced since a source node may send a lot of ants in the route setup process. Also, it does not use additional techniques to determine the congestion state of the route because it is based on the estimated delay and on the data load information. Another drawback of this protocol is that it does not use mechanisms that can help to discover shorter routes to improve the algorithm convergence.

AMQR [10] is a proposal based on multipath techniques. Most of the routing protocols based on ants for MANETs are essentially routing methods of single route, which tend to have an overhead on the nodes, which are on the shortest way from the origin to the destination. This overhead is due to the fact there is no load balancing in the methods about unique path. The Disjoint-link multipath routing is more robust and can guarantee a best QoS than unique path routing of the MANETs. AMQR combines swarm intelligence and Disjoint-link multipath to balance the load. It establishes and utilizes Disjoint-link multipath to send data packets and to adapt the pheromone to disperse the communication traffic

Protocol [11] is a variant of AODV [12] and it is based on ACO techniques at the time of the link failure repair process, improving the delivered packet ratio, the throughput and average End-to-End delay. Moreover it reduces the overhead because it is based on control packets that are forward and backward ants.

As mentioned, this protocol gets greater efficien y than the AODV algorithm by using the properties of the ACO algorithms in the local link repair. In the original version AODV when there is a link failure, the node that detects this failure generates a route error message (RERR), which is sent to the source of the data session in a hop-by-bop manner. The origin performs a new route setup when this node receives this message. This causes a high overload, when there is a high mobility of the nodes. This new variant of AODV has a link failure management based on findin alternate routes using the function of these agents which reduce the overhead.

However, this reactive approach does not take advantage of the hybrid algorithms, having therefore a high latency in the route setup process.

To get the advantage of both approach, Hybrid protocols exists. They are a combination between proactive and reactive. We show some representative hybrid protocols proposed in the literature.

Ant-AODV [13], hybrid routing protocol based on ACO and on the routing protocol AODV, as its name suggests, it tries to take advantage of both. To overcome some of the disadvantages of AODV, as is the overhead generated by the increase of control message, this hybrid technical is utilized, that highlights the node connectivity and decreases the End-to-End delay, as well as the latency of route setup process. This protocol was designed without taking into account techniques to help to fin the shortest routes and mechanisms to mitigate the congestion problem.

HOPNET [14] is based on a technique in which the ants jump from one zone to another one. The algorithm has characteristics extracted from the ZRP and DSR protocols, being highly scalable, compared with other hybrid protocols. This algorithm consists of proactive route setup in the area of node vicinity, and communication between zones reactively on demand is done when it sends packets from a zone to

another. When the number of nodes is small the continuous movement of peripheral nodes constantly attempts to discover new routes, which causes more delay than in other hybrid routing protocols.

Another protocol is [15], which combines ideas about ACO with Zone-based Hierarchical (ZHLS) protocol. Its algorithm is similar to HOPNET and it is based on ants which cross from one zone to the next one. The authors claim that their proposal improves the performance with regard traditional algorithms, according to the delay, ratio and overhead metrics.

But undoubtedly the most representative is AntHocNet [16,17]. It constitutes a hybrid, adaptive and multipath protocol that takes into account the dynamic topology and other characteristics of the MANETs, presenting a hybrid mode of operation: it is reactive because it has agents operating in the route setup to destinations and proactive due to other agents collecting information to discover new routes in the prevention against link failure. It is multipath because it establishes different paths to send the information to the destination. Finally, it is adaptive because it suits the traff c and network conditions.

On the other hand, a work related to immune systems is [18]. In this work the authors try to solve problems of misbehaviors in mobile ad hoc networks (MANETs) taking into account the artificia immune systems, but they have used the standard protocol DSR which it is reactive and it does not exploit the properties of the hybrids.

3 AntOR and its new enhancements

In this section we review a novel protocol which is based on AntHocNet, concretely in Ducatelle's Thesis [17]. The hybrid (mix between reactive and proactive part) routing protocol AntOR [3,4] has the following different characteristics which are different from AntHocNet:

- Disjoint-link and disjoint-node protocol [19].
- Separation between the pheromones values in the diffusion process.
- Use of the distance metric in the proactive path exploration.

AntOR provides two versions in its design: the disjoint-link (AntOR-DLR) in which the links are not shared and disjoint-node (AntOR-DNR) in which the nodes are not shared. Every disjoint-node is also a disjoint-link, but not vice versa. Both types of disjoint routes have the following advantages:

1. A failure in one node only affects a path, not the entire network.
2. Load balancing is better because there are not repeated routes on the disjoint property.

However, the use of such routes needs more resources by not sharing the links or nodes.

Figure 1 shows an example of disjoint-link routes, where it is observed how the constitution of disjoint-link routes is in regard to no disjoint-link.

The basic idea for findin and representing disjoint-link disjoint routes is to mark each link disjoint with a label indicating what the origin the data session origin is. In

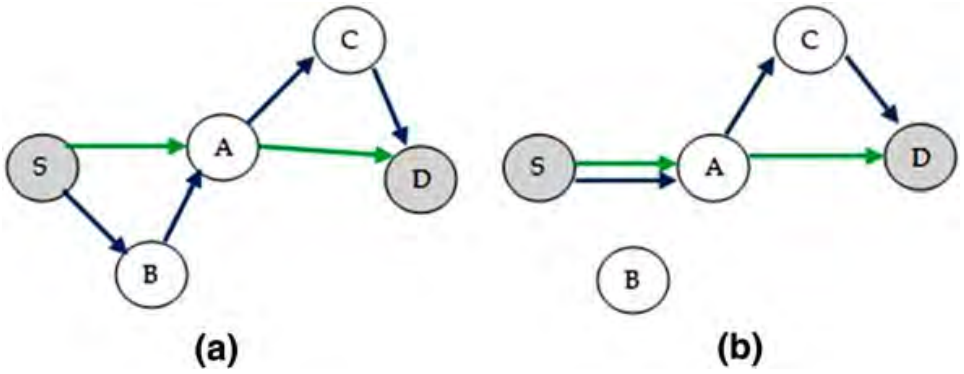


Fig. 1 Disjoint-link routes (**a**) versus no disjoint-link routes (**b**)

this example, case a of Fig. 1 corresponds to disjoint-link routes and case b to non-disjoint routes. As seen in case a of Fig. 1 the node A is shared by main route (green) and the alternate route (blue), is therefore disjoint-node but not disjoint-link. This is the reason why every disjoint-node is also a disjoint-link,but not vice versa.

One advantage of disjoint-link is its good management to a link failure. Specificall in case a if the link (B, A) of blue route fails, the data could still be transmitted by the main route [S, A, D]. In contrast, in case b, if node A fails affect many routes, being two of them overlapped (S, A) so it would harm the protocol performance.

This algorithm is applied to mobile ad hoc networks (MANETs) and the optimization might be addressed by ideas of immunology. This algorithm is modeled in the following way:

- **Body**: The entire mobile ad hoc network.
- **Antibody**: Address pairs consisting of the "next hop" and "destination".
- **Antigen**: Destination of the data packet.
- **Matching**: Correspondence between the associated destination with the data packet and destination fiel of a pair which it belongs to an antibody.
- **Affinity**: Heuristic value (Regular pheromone).

We present new enhancements applied to AntOR-DLR. We would rather AntOR-DLR than AntOR-DNR because of results obtained in the experimentation [19].

These new techniques, used and reflecte in the simulation results, are: control packet buffering, outdated message management, failure link management, route exploration management.

3.1 Control packet buffering

This new technique is buffering properties in which the control packets are stored. Every time interval around 100 ms, those packets are sent to corresponding destinations. The information included in each entry which is stored in this buffer is: (a) socket to send the packet, (b) the control packet which is the particular message of the protocol and (c) destination address (it could be a broadcast address or a unicast address sent to a determined node).

3.2 Outdated message management

This routing approach utilizes a new technique to control the outdated messages. It is a method that replaces to pheromone evaporation. This event each time interval of 2 s is realized. The process is the following:

- The timestamp fiel with the current time of process is established, every time a register in the routing table is updated or created.
- If the event occurs and the timestamp fiel associated an each route from routing table is lower than current time minus a time limit (it is established to 5 s in the experiments), the particular register of the route is deleted from the routing table.
- This time limit can vary according to implementations. If it has low values, then the system converges slowly to premature routes, but with the drawback that the system may erase routes to active destinations. On the other hand, if it has high values, it implies a high convergence in the creation of the routes but with disadvantage of keeping outdated routes in the system.

3.3 Failure link management

Another technique is related to fault tolerant. When a fault in a control message (an agent of our algorithm) is detected, we trigger a mechanism of neutralization process. This makes that, in highly dynamic environments, we have to trigger more neutralization mechanisms by sending agents to repair the route or notify to the precursors of the node that detects the failure until reaching the source of the data session indicating that the route is disconnected. This implies an overhead in packets and bytes. To f x this we use a new technique that checks if exists route (regular pheromone value greater than zero) to the neighbor whose we want to transmit, seeing this information in the routing table. If path exists, we send the control message, otherwise the agent does not send. Thus, this prevents the failure neutralization and it reduces overhead.

3.4 Route exploration management

Reduction of overhead of the system through proactive agents that do not need virtual pheromone routes. These agents create alternative routes and go from neighbor to neighbor until reaching the destination node. At the time of selecting the next hop they take into account the maximum value of regular pheromone to such a one-hop neighbor. Alternative routes are achieved with this technique up to a limit which is selected previously, and which are disjoint because those routes do not belong to the main route.

Figure 2 shows an example about the selection of the next hop to forward in proactive process.

Firstly we note the main route, created in the route setup process, is [A, B, E]. The node A needs to forward the corresponding proactive agent or ant and to do this, it has to choose among three intermediate neighbors: B, C, D. These neighbors have pheromone values of 20, 5, and 15 respectively because of the pheromone value

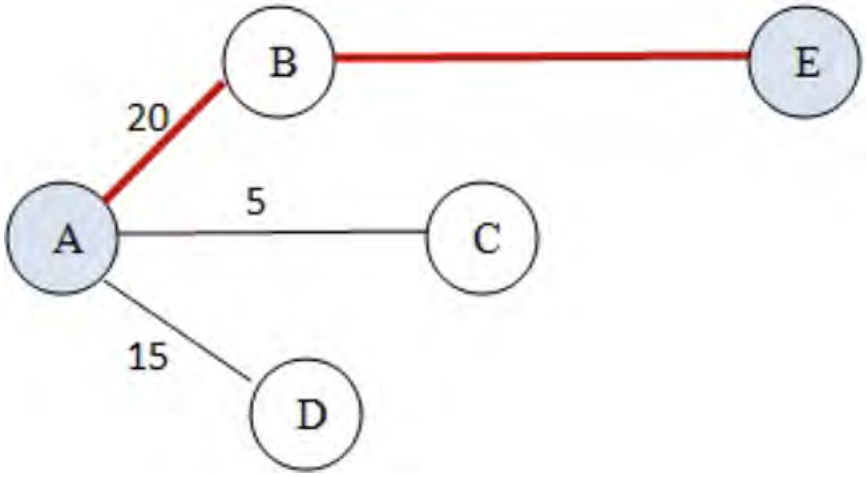


Fig. 2 Example of proactive process

is inversely proportional to cost based on the distance between one-hop neighbors. According to our technique, the best candidate to forward is B (regular pheromone value of 20), but node A realizes itself that node B belongs to main route [A, B, E]. Then node A only can choose the intermediate node D (regular pheromone value of 15) to send the proactive ant. This process continues across other intermediate nodes until reaching the destination node.

This algorithm, AntOR-DLR, with the previous techniques is called AntOR-v2, then the simulation results is presented in the next section.

4 Simulation results

The most important performance metrics that were used in the experimentation are:

- **Jitter**: performance metric that measures the delay variation between consecutive packets received.
- **Delivered data packet ratio**: relationship between number of packets sent and the number of packets delivered successfully.
- **Throughput**: Volume of work or information f owing through a system. It is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.
- **Overhead in number of packets**: relationship between the total numbers of transmitted control packets by the nodes of network and the number of delivered data packets to their destinations.

The characteristics of the simulations in network simulator NS-3 were: we used 100 nodes randomly distributed and configure with a transmission range of 300 m. The nodes are moved according to the Random WayPoint (RWP) pattern, varying pause time from a low of 0–240 s at intervals of 60 s. The scenario was rectangular with dimensions 3000 m × 1,000 m. The speed was variable from a minimum of 0–8 m/s. It used 10 random data sessions using the application protocol constant bit rate (CBR) beginning to send data at random from 0 s to a maximum of 60 s. The sending rate

was 512 bit/s, i.e., sending a packet of 64 bytes per second. The maximum simulation time was established to 300 s. It employed a total of 10 runs in the experiment.

Increasing the pause time has two different effects on the general properties of the scenario that are relevant for routing. The f rst of these is a decrease in node mobility: since nodes stay still for longer periods, they are less mobile, and the network becomes less dynamic. As a consequence, the scenario becomes less difficult The second one is a bit less straightforward, and has to do with the distribution of nodes over the network area when the RWP mobility model is used.

Figure 3 shows how Jitter has a better behavior in AntOR-DLR than in AODV approach.

Figure 4 shows how the data packet ratio in our proposed protocol AntOR-DLR is better than in AODV at all time. The ratio is an important metric of effectiveness.

Figure 5 shows that throughput for both approach is similar than Fig. 3 but using another scale.

Figure 6 shows how the overhead in number of packets in AntOR is practically the same as AODV.

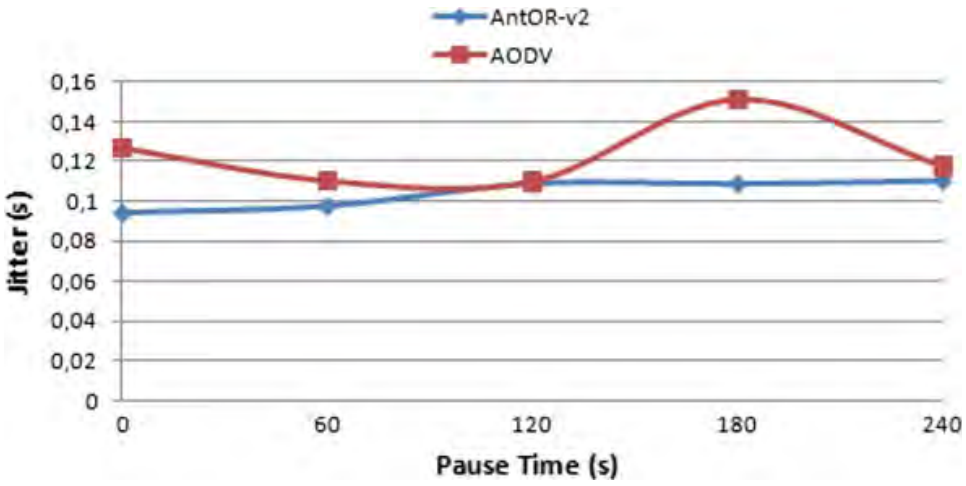


Fig. 3 Jitter

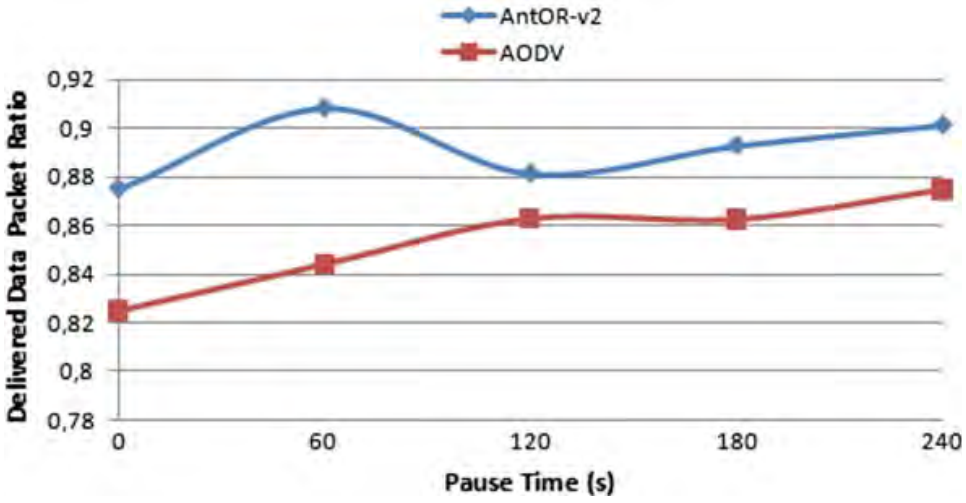


Fig. 4 Delivered data packet ratio

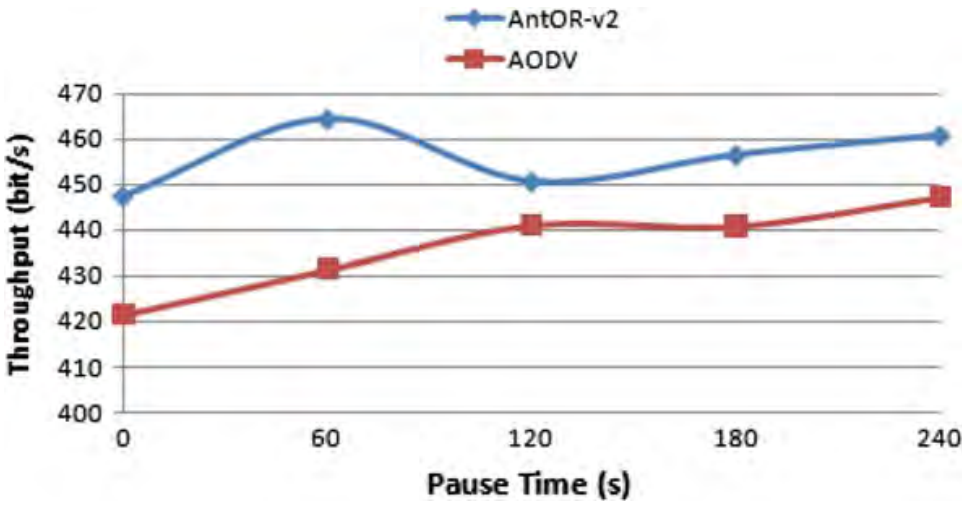


Fig. 5 Throughput

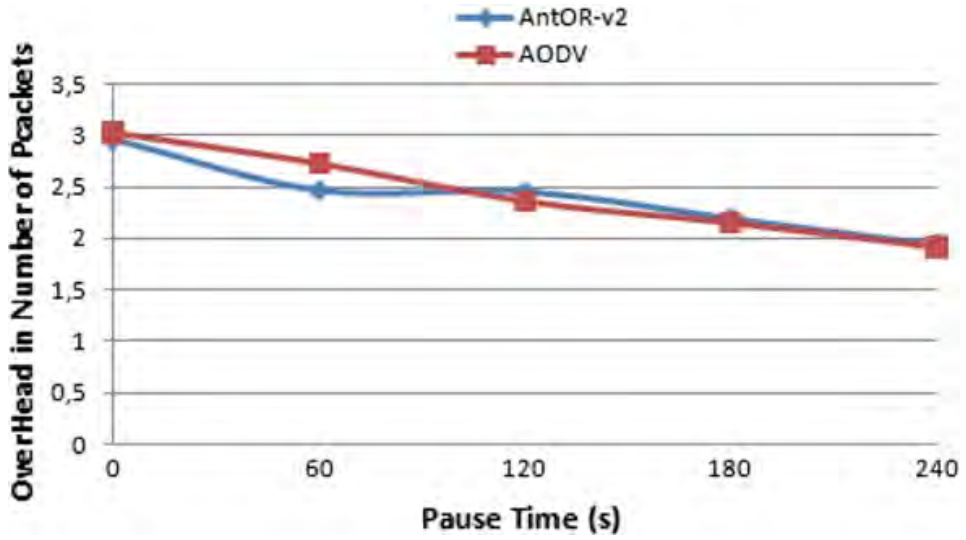


Fig. 6 Overhead in number of packets

5 Conclusion

In this article we have seen that the optimization problem are very revelant our daily lives.

In particular, we have considered the routing problem. It consists on forwarding the data from the sources to destinations in a multihop manner. To this end, we have reviewed an ACO bioinspired algorithm, called AntOR, which is applied to mobile ad hoc networks. Due to the highly dynamic environments of these networks, a robust and efficien protocol needed.

Also we have associated our protocol with concepts of artificia immune systems to prove the existence of a direct relationship each other.

On the other hand, we have modifie the original protocol AntOR, providing new optimization techniques as control packet buffering, outdated message management, failure link management, route exploration management.

Finally we can see that with these new techniques, called AntOR-v2, we get better results according to metrics of jitter, delivered data packet ratio, throughput and overhead in number of packets. In this experimentation we have compared AntOR-v2 with purely reactive protocol AODV, well-known in the literature.

Acknowledgments This work was supported by the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID MEDITERRÁNEO A1/037528/11. This work was also supported by the Security Engineering Research Center, granted by the Ministry of Knowledge Economy (MKE, Korea).

References

1. Dorigo M (1992) Optimization, learning and natural algorithms. Doctoral Thesis, Politecnico di Milano, Italie
2. Kennedy J (2001) Swarm intelligence. Morgan Kaufmann Publishers, Burlington
3. García LJ, Rupérez D, Sandoval AL (2010) Bioinspired routing protocol for mobile ad hoc networks. IET Commun 4(18):2187–2195
4. Rupérez Cañas D, Sandoval Orozco AL, Kim TH (2011) Comparing AntOR-disjoint node routing protocol with its parallel extension. Commun Comput Inf Sci (CCIS) 263:305–309
5. Ramanathan R, Redi J (2002) A brief overview of ad hoc networks: challenges and directions. Commun Magazine IEEE 40:20–22
6. Baras JS, Mehta H (2003) A probabilistic emergent routing algorithm for mobile ad hoc networks, modeling and optimization in mobile ad hoc wireless networks WiOpt' 03, March
7. Hossein O, Saadawi T (2003) Ant routing algorithm for mobile ad hoc networks (ARAMA). In: Proceedings of the 22nd IEEE International Performance, Computing, and Communications Conference. Phoenix , Arizona, USA, pp 281–290
8. Günes M, Sorges U, Bouazizi I (2002) ARA—The ant-colony based routing algorithm for MANETs. In: Proceedings of the ICPP International Workshop on Ad Hoc Networks (IWAHN)
9. Zheng X, Guo W, Liu R (2004) An ant-based distributed routing algorithm for ad-hoc networks, International Conference on Communications. Circuits Syst, ICCCAS 2004, 1(1), 412–417, 27–29
10. Liua L, Feng G (2005) A novel ant colony based QoSAware routing algorithm for MANETs, ICNC 2005, LNCS 3612, Springer, Berlin, pp 45766
11. Jain J, Gupta R, Bandhopadhyay TK (2011) Ant colony algorithm in MANET-local link repairing of AODV, Electronics Computer Technology (ICECT), 2011 3rd International Conference on 6, 270–273, 8–10
12. Perkins CE, Belding-Royer EM, Das S (2003) Ad hoc on-demand distance vector (AODV) routing. RFC3561, July http://tools.ietf.org/html/rfc3561
13. Marwaha S, Tham CK, Srinavasan D (2002) Mobile agents based routing protocol for mobile ad hoc networks. In: IEEE Global Telecommunications Conference (GLOBECOM'02). Taipei, Taiwan
14. Wang J, Osagie E, Thulasiraman P, Thulasiram RK (2009) HOPNET: a hybrid ant colony optimization routing algorithm for mobile ad hoc network. Ad Hoc Netw (Elsevier Science Publishers) 7(4):690–705
15. Rafsanjani MK, Asadinia S, Pakzad F (2010) A hybrid routing algorithm based on ant colony and ZHLS routing protocol for MANET. FGIT-FGCN (2) 120:112–122
16. Di Caro G, Ducatelle F, Gambardella LM (2004) AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks. In: Proceedings of PPSN VIII—Eight International Conference on Parallel Problem Solving from Nature, Birmingham, UK, Springer, Lecture Notes in Computer Science 3242:18–22
17. Ducatelle F (2007) Adaptive routing in ad hoc wireless multi-hop networks, PhD thesis, Università della Svizzera Italiana, Istituto Dalle Molle di Studi sull'Intelligenza Artificial
18. Le Boudec J, Sarajanoviè S (2004) An artificia immune system approach to misbehavior detection in mobile ad-hoc networks. In: Proceedings of Bio-ADIT 2004 (The First In-ternational Workshop on Biologically Inspired Approaches to Advanced Information Technology), Lausanne, Switzerland, pp. 96–111, January 29–30
19. Rupérez D, Sandoval AL, García LJ, Kim TH (2011) A comparison study between AntOR-disjoint node routing and AntOR-disjoint link routing for mobile ad hoc networks. Commun Comput Inf Sci (CCIS) 263:300–304

AN ANT-BASED ADAPTIVE DISTRIBUTED ROUTING PROTOCOL FOR MOBILE AD HOC NETWORKS

Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco, Luis Javier García Villalba

Group of Analysis, Security and Systems (GASS)
Department of Software Engineering and Artificial Intelligence (DISIA)
School of Computer Science, Office 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain
Email: {delfinrc, asandoval, javiergv}@fdi.ucm.es

Abstract

Mobile ad hoc network (MANET) is a network in which all nodes are mobile and communicate exclusively via wireless connections. There is no fixed infrastructure in the network, and there is no hierarchy. Routing is the task of directing data flows from source nodes to destination nodes while maximizing network performance. Due to the ad hoc and dynamic nature of the network, the topology can change constantly, and paths between sources and destinations that were initially efficient can quickly become inefficient or even infeasible. In this work, we present an adaptive distributed routing protocol for mobile ad hoc networks based on Ant Colony Optimization (ACO). The simulation results show how this new protocol is better than AODV protocol.

Keywords - Mobile Ad Hoc Network, MANET, Routing, Protocol, ACO, Swarm Intelligence.

1 INTRODUCTION

One of the most important developments in recent years in the field of telecommunication networks is the increased use of wireless communication. A wide range of different wireless technologies and standards have been developed, including Wireless-Fidelity [1] (WiFi, IEEE 802.11), Bluetooth [2] (IEEE 802.15.1), Zigbee [3] (IEEE 802.15.4), Ultra Wide Band [4] (UWB, IEEE 802.15.3), Worldwide Interoperability for Microwave Access [5] (WiMax, IEEE 802.16), etc. These technologies are being made available on an ever increasing number of devices such as laptops, mobile phones, palmtops, etc., allowing them to connect to a variety of different networks. This explosive growth has made wireless communication networks one of the most important areas of research in computer science.

Hence, in recent years [6], a growing number of devices are getting equipped with networking capabilities. Many of these devices are mobile and communicate using a variety of wireless technologies, which allow them to connect to existing telecommunication networks and to each other. One can then combine a number of such devices with minimal planning to form a network.

Mobile ad hoc networks (MANETs) [6] are networks in which all nodes are mobile and communicate exclusively via wireless connections. Usually, the nodes are equipped with a single, omnidirectional wireless antenna. There is no fixed infrastructure in the network, and there is no hierarchy: all nodes are in principle equal, and can function both as end points of data communication, and as routers, forwarding data for each other in multi-hop fashion.

Routing is the task of directing data flows from source nodes to destination nodes while maximizing network performance. Due to the ad hoc and dynamic nature of the network, the topology can change constantly, and paths between sources and destinations that were initially efficient can quickly become inefficient or even infeasible. This means that routing information should be updated more regularly than in traditional wired telecommunication networks. However, this can be a problem in MANETs, with their limited bandwidth and node resources, and their possibly unreliable communication channels. New routing algorithms are therefore needed, which can give adaptivity in an efficient and robust way[7].

There are several routing approaches, so-called traditional algorithm (OLSR [8], AODV [9]), even though they are valid for routing, something these ones does not offer expected results according to analyzed performance metrics. For this reason, the researchers are focused on natural behavior of some animals (the most of them are insects) to solve complex problems. These kinds of techniques are called bioinspired algorithms and they can solve computational problems in an efficient manner.

There is a particular type of these algorithms that treats about the behavior of the ants at the time of obtaining the food. Ant Colony Optimization (ACO) [10][11] is proposed by Dorigo in his Thesis [12] and it has much influence to solve problems, such as the routing. Moreover, ACO is also based on Swarm Intelligence [13], in the collective behavior of the animals.

In this work we present a proposal of a ACO routing protocol. This paper is divided into 4 sections with the first section this introduction. In section 2 we discuss the most relevant hybrid related work in the routing based on ACO. In section 3 we present our proposal, explaining its major characteristics and analyzing the simulation results. Finally we offer conclusions in section 4.

2 RELATED WORKS

The ACO-based routing protocol can be classified, as well as the traditional protocol, in proactive, reactive and hybrid.Proactive protocols frequently need to exchange packets between mobile nodes and continuously to update their routing tables. It leads a lot of overhead. To avoid this, the reactive protocols appear, but they have more latency. This protocols act on-demand, they sent reactive agents only when needed, i.e., when one node has active data session and the node is prepared to send the data.To get the advantage of both approaches, there are hybrid protocols. They are a combination between proactive and reactive.

There are some representative hybrid protocols proposed in the literature: Ant-AODV [14], HOPNET [15], ZHLS [16], etc. But undoubtedly the most representative is AntHocNet [17] [7]. It constitutes a hybrid, adaptive and multipath protocol that takes into account the dynamic topology and other characteristics of the MANETs, presenting a hybrid mode of operation: it is reactive because it has agents operating in the route setup to destinations and proactive due to other agents collecting information to discover new routes in the prevention against link failure. It is multipath because it establishes different paths to send the information to the destination. Finally, it is adaptive because it suits the traffic and network conditions.

Finally, AntOR [18] is a protocol based on AntHocNet but it differs from this in the following characteristics: i) it is a protocol that works in two separate modes: Disjoint-link and Disjoint-node; ii) it takes into account the pheromone separation in the diffusion process; iii) Use of the distance metric in path exploration. In such protocol there are two kinds of routes: Disjoint-node and Disjoint-link. The first corresponds to routes in which nodes are not shared and the latter refers to routes in which links are not shared.

3 OUR PROPOSAL

In this work, we present a new hybrid bioinspired protocol. Main characteristic is that it belongs to hybrid algorithms because it is a combination between proactive and reactive parts:

- Reactive: It acts on-demand sending reactive agents or ants for routing setup process, when there is available data packet to be sent toward destination.
- Proactive: In the path exploration process, the source node of the data session sends proactively, in time intervals, agents for creation of alternatives routes. This process only occurs when the communication between source and destination has succeeded during the reactive process.

Moreover it operates in a multipath way because it establishes different paths to send the information to the destination. Also, it is adaptive because it suits the traffic and network conditions. It has the following enhancements: Control Packet Buffering, Outdated Route management, Data Packet Management, Link Failure Management, and Route Exploration Management. These enhancements are very important and constitute the main core in this protocol.

We have performed several tests with the Network Simulator NS-3. We have compared our proposal with purely reactive protocol AODV. We vary the pause time from 0 s to 240 s using a time interval of 60 s. We use randomly distributed 100 nodes with transmission range of 300 m. The nodes are moved according to the Random Way Point (RWP) pattern. The scenario was rectangular with dimensions 3000 m x 1000 m. The speed is constant with value of 5 m/s. It use 10 random data sessions using the application protocol Constant Bit Rate (CBR) beginning to send data at random from 0 s to a maximum of 180 s. The sending rate is 512 bit/s, i.e., sending a packet of 64 bytes per second. The maximum simulation time is established to 900 s. It employs a total of 10 runs in the experiment.

Fig. 1 shows how our proposal has a better delay than AODV at all pause time. We can note that in our protocol the ratio never reaches the value of 100 ms.Increasing the pause time has two different effects on the general properties of the scenario that are relevant for routing.Firstly, there is a decrease in node mobility: since nodes stay still for longer periods, they are less mobile, and the network becomes less dynamic. As a consequence, the scenario becomes less difficult. Secondly it is related to the distribution of nodes over the network area when the RWP mobility model is used.

Regard to overhead in bytes, in Fig. 2 we appreciate how the overhead in our proposal is lower than in AODV at all pause time. Also we show that the curve of our protocol does not have extreme behaviour and it is practically a straight line.

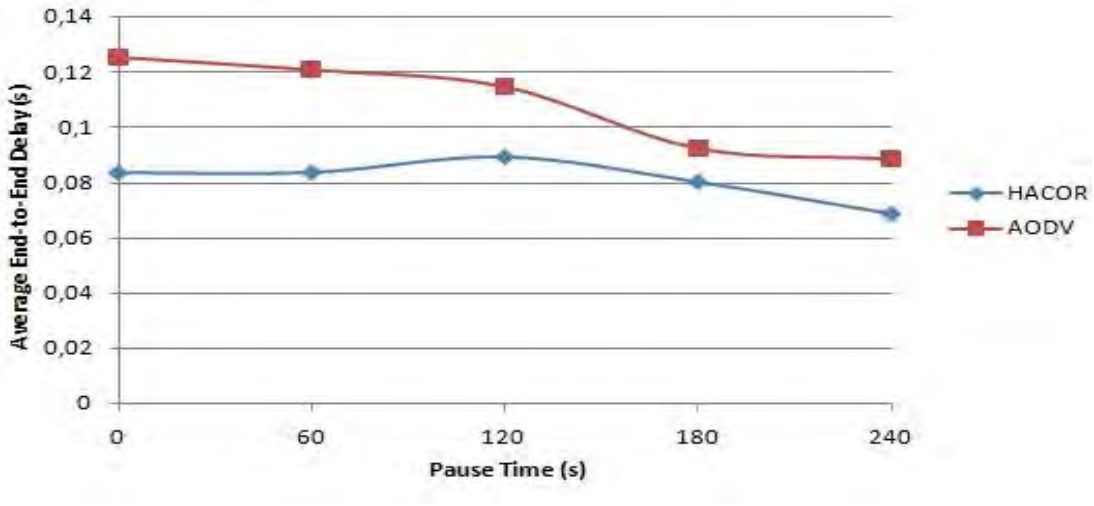


Fig.1 Average End-to-End Delay

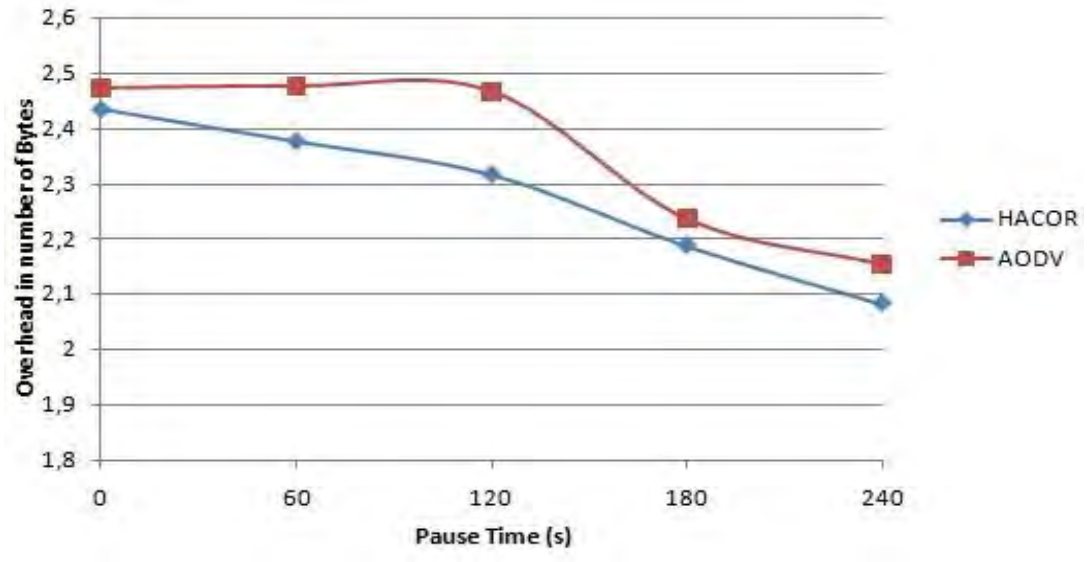


Fig.2 Overhead in Number of Bytes

4 CONCLUSIONS

In this paper we present a hybrid ACO-Based routing protocol which has novel characteristics such as control packet buffering, outdated route management, data packet management, failure link management, and route exploration management.The experimentation results show that this new protocol has a better behavior than AODV according to analyzed metrics.

ACKNOWLEDGMENTS

This work was supported by the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID MEDITERRÁNEO A1/037528/11.

References

[1] IEEE 802.11 Standard Group Website. Available from: http://www.ieee802.org/11/.

[2] Bluetooth Special Interest Group. Specification of Bluetooth System, Version 1.1, February 2002.

[3] The Zigbee Alliance. Available from: http://www.zigbee.org/en/index.asp.

[4] IEEE 802.15 Standard Group Website. Available from: http://www.ieee802.org/15/.

[5] IEEE 802.16 Standard Group Website. Available from: http://www.ieee802.org/16/.

[6] R. Ramanathan, J.Redi: A Brief Overview of Ad Hoc Networks: Challenges and Directions. IEEE Communications Magazine, Vol. 40, pp. 20-22, 2002.

[7] F. Ducatelle: Adaptive Routing in Ad Hoc Wireless Multi-hop Networks.PhD Thesis, UniversitàdellaSvizzera Italiana, Istituto Dalle Molle di Studisull´IntelligenzaArtificiale, 2007.

[8] T. Clausen, P.Jacquet: Optimized Link State Routing Protocol (OLSR). IETF RFC3626, October 2003.Available from: http://www.ietf.org/rfc/rfc3626.txt.

[9] C. E. Perkins, E. M. Belding-Royer, S. Das: Ad Hoc On-Demand Distance Vector (AODV) Routing. IETF RFC3561, July 2003.Available from: http://tools.ietf.org/html/rfc3561.

[10] M. Dorigo, T. Stützle: Ant Colony Optimization. The MIT Press, MA, USA, 2004.

[11] G. A. Di Caro: Ant Colony Optimization and its Application to Adaptive Routing in Telecommunication Networks. PhD Thesis in Applied Sciences, Polytechnic School, UniversitéLibrede Bruxelles, Brussels, Belgium, 2004.

[12] M. Dorigo: Optimization, Learning and Natural Algorithms. PhD Thesis, Politecnico di Milano, Italy, 1992.

[13] J. Kennedy: Swarm Intelligence. Morgan Kaufmann Publishers, 2001.

[14] X. Zheng, W. Guo, R. Renting Liu: An Ant-based Distributed Routing Algorithm for Ad-Hoc Networks.Proceedings of the International Conference on Communications, Circuits and Systems, pp. 412-417, June 2004.

[15] J. Wang, E. Osagie, P.Thulasiraman, R. K. Thulasiram: HOPNET: A Hybrid Ant Colony Optimization Routing Algorithm for Mobile Ad Hoc Network. Ad Hoc Networks, Vol. 7, No. 4, pp. 690-705, 2009.

[16] M. K. Rafsanjani, S. Asadinia, F. Pakzad: A Hybrid Routing Algorithm Based on Ant Colony and ZHLS Routing Protocol for MANET.Proceedings of the International Conference Communication and Networking (FGCN 2010), Communications in Computer and Information Science (CCIS), Vol. 120, pp. 112-122, 2010.

[17] G. Di Caro, F. Ducatelle, L. M. Gambardella: AntHocNet: An Ant-based Hybrid Routing Algorithm for Mobile Ad Hoc Networks. Proceedings of the Eight International Conference on Parallel Problem Solving from Nature (PPSN VIII), Birmingham, UK, Lecture Notes in Computer Science (LNCS), Vol. 3242, pp. 461-470, September 2004.

[18] L. J. García Villalba, D. RupérezCañas, A. L. Sandoval Orozco: Bioinspired Routing Protocol for Mobile Ad Hoc Networks, IET Communications, Vol. 4, No. 18, pp. 2187-2195, 2010.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 265485, 7 pages
http://dx.doi.org/10.1155/2013/265485

Research Article

Hybrid ACO Routing Protocol for Mobile Ad Hoc Networks

D. Rupérez Cañas,[1] A. L. Sandoval Orozco,[1] L. J. García Villalba,[1] and P.-S. Hong[2]

[1] *Group of Analysis, Security and Systems (GASS), Department of Software Engineering and Artificial Intelligence (DISIA), School of Computer Science, Office 431, Universidad Complutense de Madrid (UCM), Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain*
[2] *School of Information Technology, Sungshin Women's University, 249-1 Dongseon-dong 3-ga, Seoul 136-742, Republic of Korea*

Correspondence should be addressed to L. J. García Villalba; javiergv@fdi.ucm.es

Received 19 April 2013; Accepted 7 May 2013

Academic Editor: Sabah Mohammed

Mobile ad hoc networks (MANETs) are formed by wireless mobile devices that do not require a predefined infrastructure and where all nodes are at the same level without the need of a central coordinator. The routing is an important task in this kind of networks because of highly dynamic environments and other characteristics. Routing consists of sending the information from sender nodes to destination nodes in a multihop manner. There are different techniques to solve the routing problem. We rely on bioinspired algorithms, those that take into account the behavior of certain animals to solve it. Specifically we design a novel routing algorithm based on Ant Colony Optimization (ACO) which uses the concepts of Swarm Intelligence and analyzes the behavior of ants at the time of obtaining the food. In this work, we present Hybrid ACO Routing (HACOR), a bio-inspired protocol based on ACO. In the simulation results, we observe how this protocol performs significantly better compared to a state-of-the-art routing protocol, according to the analyzed metrics.

1. Introduction

Nowadays the wireless networks are having a relevant importance due to the services offered by the high proliferation of the devices. Mobile ad hoc networks (MANETs) [1] are constituted by wireless mobile devices/nodes distributed without the need of predefined infrastructure; that is, each node has the same level in the network and they can act as client or server. The nodes may join or depart the network at any time, and thus wireless links are constantly created and destroyed.

The main problems of this kind of networks are autoconfiguration and routing. The autoconfiguration of mobile ad hoc networks requires specific protocols [2]; that is, traditional solutions are invalid. Similarly, routing in these networks requires particular solutions, like in wireless sensor networks [3]. Some protocols [4] combine autoconfiguration and routing aspects to optimize the operation of such networks.

Considering the limited communication range of wireless networks, information is usually transmitted in a multihop fashion. Data routing is an essential task in networking and it has the mission of finding routing paths from sender to destination nodes. In highly dynamic environments, such as MANETs, this task becomes particularly challenging, and routing protocols must be designed to cope with the time-varying topologies.

There are several routing approaches for MANETs. The so-called traditional algorithms (OLSR [5], AODV [6]), even though they are widely used for routing, usually do not meet the expectation in terms of performance. For this reason, the researchers have focused on natural behavior of some animals (the most of them are insects) to solve complex problems. These kinds of techniques are typically referred to as *bioinspired algorithms*.

Ant Colony Optimization (ACO) [7, 8] is a particular type of these algorithms which takes inspiration from the behavior of ants at the time of obtaining the food. ACO is a metaheuristic proposed by Dorigo in his thesis [9], and it has been used to solve different type of problems, including network data routing.

In this work we present Hybrid ACO Routing (HACOR) protocol. It is a novel ACO-based algorithm which combines proactive and reactive parts. HACOR proposes several enhancements and additional functionalities in comparison to previous ACO-based algorithms. Within these, a novel buffering technique for control packets, an improved link failure management procedure, and a proactive route exploration mechanism exist. These strategies were carefully designed for MANETs, aiming to reduce the protocol overhead and improve the network performance in terms of end-to-end delays and data throughput.

This paper is divided into 5 sections with the first section being this introduction. In Section 2, we discuss the most relevant hybrid related work in the routing based on ACO. In Section 3, we present our proposal, HACOR, explaining its major characteristics. In the following Section 4, we analyze the simulation results where HACOR and AODV are compared. Finally, we offer conclusions in Section 5.

2. Related Work

The ACO-based routing protocols can be classified, as well as the traditional protocols, in proactive, reactive, and hybrid. Proactive protocols frequently need to exchange packets between nodes and to continuously update their routing tables; therefore they usually suffer from increased overhead. On the other hand, the reactive protocols act on-demand; therefore they establish routing paths and sent control packets only when needed, thus reducing overhead. This reduction comes at the cost of higher latency.

Not until recently, the community started to explore the combination of both approaches, which led to the proposal of hybrid methods. The main motivation behind the hybridization different algorithms is getting the advantages of both approaches and finding a good trade-off between overhead and latency. We discuss some representative hybrid protocols proposed in the literature. Ant-AODV [10], a hybrid routing protocol based on ACO and on the routing protocol AODV, as its name suggests, tries to take advantage of both.

However, this protocol was designed without taking into account techniques to help to find the shortest routes and mechanisms to mitigate the congestion problem.

HOPNET [11] is based on a technique in which ants jump from one zone to another one. The algorithm has characteristics extracted from the ZRP and DSR protocols, being highly scalable, compared with other hybrid protocols. This algorithm consists of a proactive route setup in the area of node vicinity and communication between zones reactively on demand, which is done at the moment of sending packets from a zone to another. When the number of nodes is small, the continuous movement of peripheral nodes constantly triggers attempts to discover new routes, which causes more overhead and transmission delays compared to other hybrid routing protocols. Another protocol is presented in [12], which combines ideas about ACO with Zone-Based Hierarchical (ZHLS) protocol. Its algorithm is similar to HOPNET and it is based on ants which cross from one zone to the next one. The authors claim that their proposal improves the performance in comparison to traditional algorithms, according to the delay, ratio, and overhead metrics. But undoubtedly the most representative is AntHocNet [13, 14]. It constitutes a hybrid, adaptive, and multipath protocol that takes into account the dynamic topology and other characteristics of the MANETs, presenting a hybrid mode of operation; it is reactive because it has agents operating in the route setup to destinations and proactive due to other agents collecting information to discover new routes in the prevention against link failure. It is multipath because it establishes different paths to send the information to the destination. Finally, it is adaptive because it suits the current traffic and network conditions.

Finally, AntOR [15, 16] is a protocol based on AntHocNet but it differs from this in the following characteristics: (i) it is a protocol that works in two separate modes: Disjoint-link and Disjoint-node; (ii) it takes into account the pheromone separation in the diffusion process; (iii) it uses a distance metric in path exploration. In this protocol there are two kinds of routes: Disjoint-node and Disjoint-link. The first corresponds to routes in which nodes are not shared and the latter refers to routes in which links are not shared.

3. Proposed Algorithm

In this work, we present HACOR which is a hybrid bioinspired protocol. Main characteristic is that HACOR belongs to hybrid algorithms because it is a combination between proactive and reactive parts.

(i) *Reactive.* It acts on-demand sending reactive agents or ants for routing setup process, when there are available data packets to be sent towards destination.

(ii) *Proactive.* In the path exploration process, the source node of the data session sends proactively, in time intervals, agents for creation of alternatives routes. This process only occurs when the communication between source and destination has succeeded during the reactive process.

Moreover, it operates in a multipath way because it establishes different paths to send the information to the destination. Also, it is adaptive because it suits the traffic and network conditions. To this end, the main functional features are as follows.

3.1. Control Packet Buffering. The motivation of this technique is the reduction of overhead. Thus, we send the control packets from the buffer synchronously. This strategy employs a buffer in which the control packets are temporarily stored. All the control packets, which are ready to send, are first stored in the control buffer. The main core of this technique consists on sending the enqueued packets to corresponding destinations in every time interval (it is established to 100 ms). The information included in each entry which is stored in this buffer is (a) socket to send the packet, (b) the control packet which is the particular message of the protocol, and (c) destination address (it could be a broadcast address or a unicast address sent to a determined node). We have utilized

```
(1) if CheckLink(dst) = true and TransmissionError(dst) = true then
(2)    DeleteNeighbor(dst);
(3)    DeleteAllRoutes(dst);
(4)    if RouteAtSource(dst) = true then
(5)       SendRFA();
(6)    else if CheckData() = true then
(7)       SendLocalRepairAnt();
(8)    else if CheckHello() = false then
(9)       SendUnicastPrecursor();
(10) end
```

Algorithm 1: Link failure management.

time intervals of 100 ms because we got the best results in the simulations. With this technique, there is not the overlapping of control packets at the time of sending.

3.2. Outdated Route Management. This routing approach utilizes a new technique to control the outdated routes. It is a method that replaces pheromone evaporation. This technique avoids the conflicts of the routes, which may be created in an unnecessary manner. To achieve this goal, an event is triggered every 2 s. The technique is described as follows. A time-stamp field with the current time is established, every time a register in the routing table is updated or created. If the associated time-stamp field of each route from routing table is minor than current time minus a time limit (it is established to 5 s in the experiments), the particular register of the route is deleted from the routing table. This time limit can vary according to implementations. If it has low values, then the system converges slowly to premature routes, but with the drawback that the system may erase routes to active destinations. On the other hand, if it has high values, it implies a fast convergence in the creation of the routes but with disadvantage of keeping outdated routes in the system.

3.3. Data Packet Management. This technique consists of checking periodically the buffer for delayed data packets. Every node, at the time of forwarding the data packet, checks if it has a route to destination. If such route does not exist, it enqueues the corresponding packet. By periodically checking the buffer and resending packets, we obtain a better delivered data packet ratio, without altering other metrics, such as delay or jitter. The time period for checking the buffer is around 100 ms. Also, we want to note that the enqueued data packets are always sent while there is a valid route to its corresponding destination. We use this technique aiming to improve the packet delivery ratio, trying not to lose any packet. To avoid affecting other metrics as average end-to-end delay or jitter, we employ a small execution time as previously mentioned.

3.4. Link Failure Management. In this management, we control two processes. (a) The first one is related to fault tolerant, and (b) the second one is related to the way of neutralization the link failure management.

A common way to handle faults in control messages (an ant agent in ACO-based protocols) is to trigger a mechanism of neutralization process. This process involves the sending of agents to repair the route or to notify the precursors of the node about the failure. These agents are propagated (by broadcasting) throughout the network until reaching the source of the data session indicating that the route has become invalid. In highly dynamic environments, such neutralization processes are executed very often, inducing a large overhead in terms of packets and bytes. To fix this weakness, we employ a new technique that checks if a route exists (i.e., regular pheromone value greater than zero). If a path exists, we send the control message (unicast); otherwise the agent does not send.

Moreover, we try to neutralize the node/link failure as follows. The first event that occurs when there is a node failure is that the node that perceives the failure updates its neighbors table and deletes all routes which have the failure node as next hop. Then we proceed in this way. If there is no route at source node, a reactive forward ant is sent. If there is no route at an intermediate node and a data packet was being forwarded when the failure occurred, a route repair forward ant is sent to every destination of all affected data sessions. If there is no route at the intermediate node and a control packet (HELLO message is not consecutively received at every certain interval) was being forwarded, no neutralization message is sent. If there is no route at the intermediate node and a unicast control packet (different to HELLO) was being forwarded, we send a unicast message to precursor node. This process is repeated as often as needed until the source node is reached. Another functionality is the next. At the moment the data packet is ready to be forwarded to next hop, our algorithm checks if there is or not a valid route to destination of the current data session. In the case there is not valid route, our algorithm enqueues the current data packet and sends a local route repair ant to fix the problem. Also, in this method, the current node sends a unicast message to all reachable neighbors. The neighbors which receive this message sent the same message to its precursors if they have it. Otherwise, they stop sending.

The basic functionality is described in Algorithm 1.

3.5. Route Exploration Management. In our algorithm, we use proactively a method in the route exploration using

the concepts of Simple-ACO [8]. With this approach, we are able to reduce even more the protocol overhead. The modifications in Simple-ACO are the following.

(i) We do not use evaporation process.

(ii) We use free-loop method when the route with all visited nodes is created.

(iii) We do not need to set initial pheromone value toward every one-hop neighbor and we do not update pheromone value anymore. We utilize the message HELLO to this end. Every node that receives a HELLO message from another one-hop neighbor updates its route to such a neighbor with the new pheromone value.

(iv) We use the probabilistic method like Simple-ACO to send the proactive Ant.

(v) We have in consideration the Disjoint-Link route in this process. Thus all alternative routes, created in this process, are disjoint. The advantage of Disjoint-Link routes are (a) a failure in one node only affects a path, not the entire network, and (b) load balancing is better because there are not repeated routes on the disjoint property.

4. Simulation Results

We have performed several tests with the Network Simulator NS-3 [17]. The most important performance metrics that were used in the experimentation are

(i) *average end-to-end delay*: measures the accumulative effectiveness of experienced delays by packets going from source to destination;

(ii) *jitter*: performance metric that measures the delay variation between consecutive packets received;

(iii) *overhead in number of bytes*: relationship between the average of transmitted control bytes and the average of delivered data bytes;

(iv) *overhead in number of packets*: relationship between the average of transmitted control packets by the nodes of network and the average of delivered data packets to their destinations;

(v) *delivered data packet ratio*: the fraction of correctly delivered data packets versus sent;

(vi) *throughput*: volume of work or information flowing through a system. It is calculated by dividing the total number of bits delivered to the destination by the packet delivery time.

We compare our proposal with purely reactive protocol AODV. For the evaluation, we use two kinds of experiments.

4.1. Experiment A. We consider 100 nodes, randomly distributed in the area, with transmission range of 300 m. The nodes are moved according to the Random Way Point (RWP) pattern. We vary the pause time from 0 s to 240 s using a time interval of 60 s. The scenario was rectangular with dimensions 3000 m × 1000 m. The speed was set to constant with value of 5 m/s. Each simulation run uses 10 random data sessions using the application protocol Constant Bit Rate (CBR) beginning to send data at random from 0 s to a maximum of 180 s. The sending rate is 512 bit/s, that is, sending a packet of 64 bytes per second. The maximum simulation time is established to 900 s. Each test consisted of 10 runs.

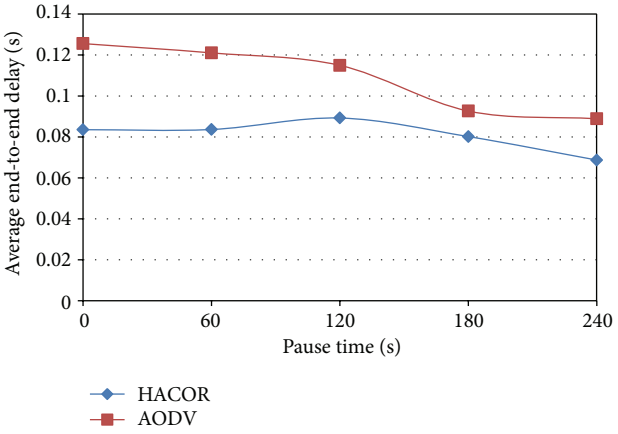


Figure 1: Average end-to-end delay (Experiment A).

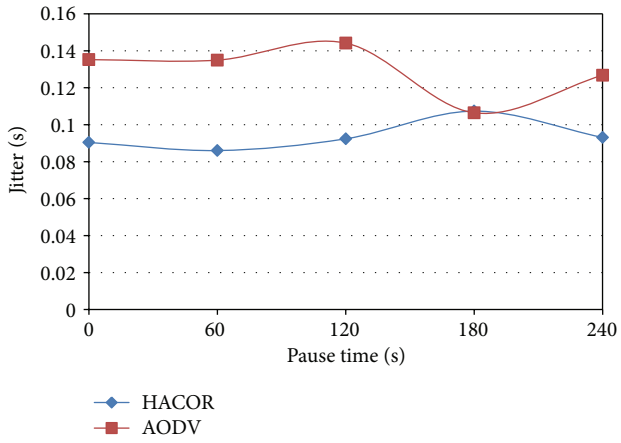


Figure 2: Jitter (Experiment A).

Figure 1 shows how our proposal HACOR has a better performance in terms of delay than AODV at all pause times considered. We can note that in HACOR the delay never reaches the value of 100 ms. Increasing the pause time has two different effects on the general properties of the scenario that are relevant for routing. Firstly, there is a decrease in node mobility; since nodes stay still for longer periods, they are less mobile, and the network becomes less dynamic. As a consequence, the scenario becomes less difficult. Secondly, it is related to the distribution of nodes over the network area when the RWP mobility model is used.

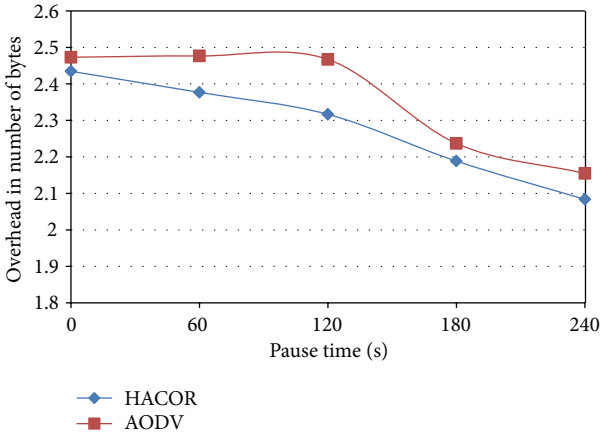


Figure 3: Overhead in number of bytes (Experiment A).

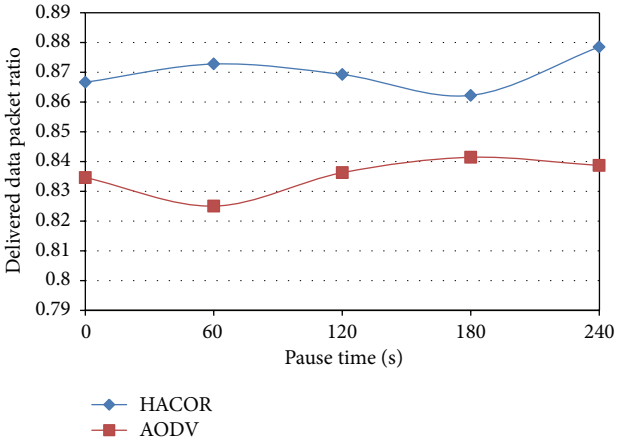


Figure 5: Delivered data packet ratio (Experiment A).

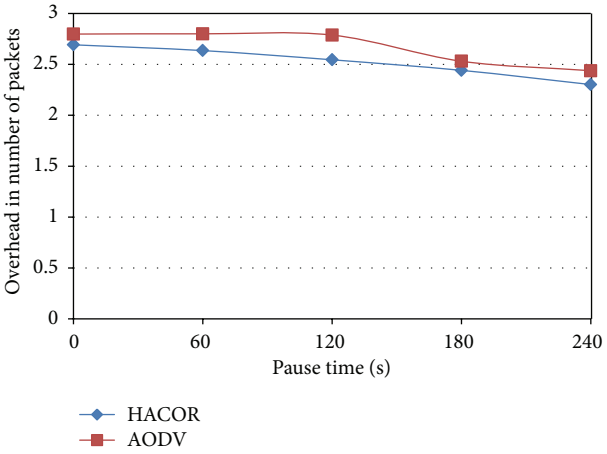


Figure 4: Overhead in number of packets (Experiment A).

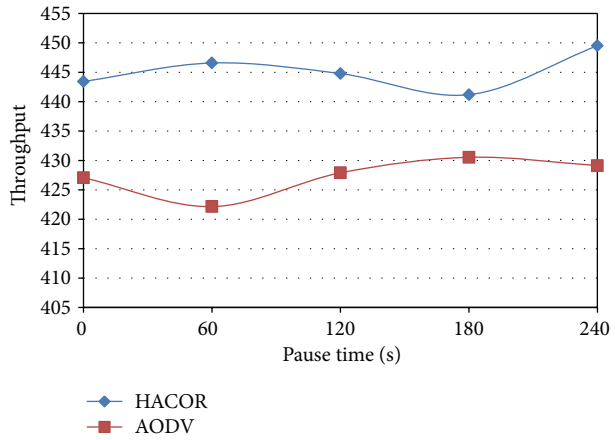


Figure 6: Throughput (Experiment A).

Figure 2 shows that the curve which represents the jitter in HACOR has a behavior more uniform than the curve in AODV. Also we can appreciate how the jitter is never more than 110 ms in our approach.

With regard to overhead in bytes, in Figure 3, we appreciate how this property in HACOR is lower than in AODV at all pause times. Also we show that the curve of HACOR does not have extreme behavior and it is practically a straight line.

Figure 4 shows a slightly similar behavior in comparison to Figure 3. In this plot, we can appreciate how the two curves are less steep. We can also observe that this overhead in number of packets is less in HACOR than in AODV.

According to Figure 5, the ratio is very distinguishing performance metric, and we observe how this metric is much higher in HACOR than in AODV at all pause times. We also check that the ratio is never less than 86%. Thus, we can affirm that we get very good results with our proposal.

Figure 6 has a behavior similar to Figure 5, but this plot uses another scale of values.

4.2. Experiment B. This experiment was designed to assess the scalability of our proposal. We vary the number of nodes in the network from a minimum of 50 nodes up to maximum of 150 nodes with a constant node density. To this end, we changed the scenario dimensions as follows: 750 m × 750 m, 875 m × 875 m, 1000 m × 1000 m, 1125 m × 1125 m, and 1250 m × 1250 m for number of nodes 50, 75, 100, 125, and 150, respectively. These nodes were randomly distributed with a transmission range of 300 m and are also moved according to the Random Way Point (RWP) pattern, which has a pause time of 2 s and speed of 5 m/s. It also uses 10 random data sessions using the application protocol Constant Bit Rate (CBR) beginning to send data at random from 0 s to a maximum of 180 s. The sending rate was 2048 bit/s, that is, sending 4 packets of 64 bytes per second. The maximum simulation time was established to 900 s. It employs a total of 10 runs in the experiment.

Figure 7 shows that delay is better in HACOR than AODV according to this complex scenario. We can observe that both curves are increasing but the curve of AODV is more accentuated than the curve of HACOR.

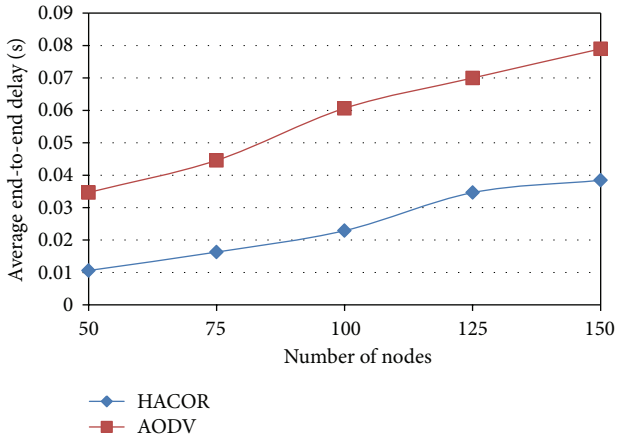


Figure 7: Average end-to-end delay (Experiment B).

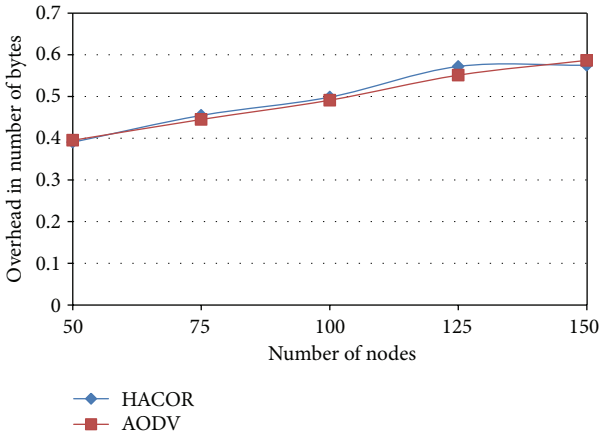


Figure 9: Overhead in number of bytes (Experiment B).

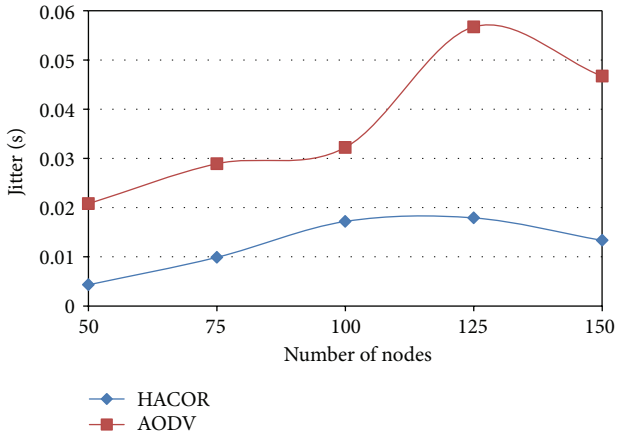


Figure 8: Jitter (Experiment B).

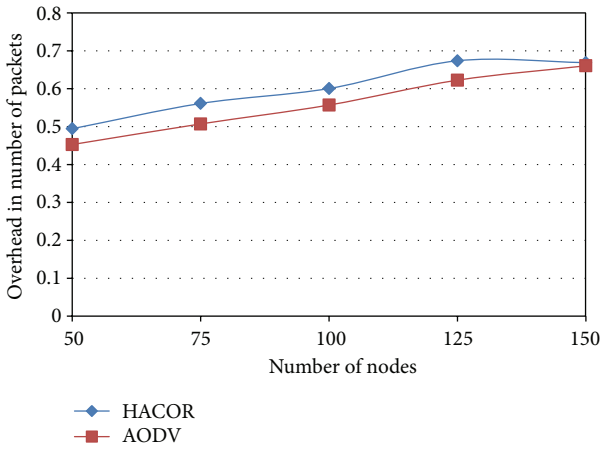


Figure 10: Overhead in number of packets (Experiment B).

Figure 8 shows how the jitter in HACOR is better than AODV, and we can appreciate the curve in HACOR having a uniform behavior than AODVs. In AODV the curve is more irregular.

Figure 9 shows the overhead in number of bytes. We appreciate that it is very similar in both approaches.

The overhead in number of packets is slightly higher in HACOR than in AODV, but in dense networks around 150 nodes (according to Figure 10) the overhead is the same. We can check a good performance with regard to this overhead in number of packets with no dense networks.

Figures 11 and 12 show the ratio and throughput, respectively. In both plots, we can appreciate the scalability of the two protocols. In Figure 11, we observe how the ratio decreases strongly in AODV than in our approach. In fact, it implies that in HACOR we obtain a better delivery ratio with the increment of nodes. Figure 12 shows the throughput but using another scale. We can appreciate a similar behavior than Figure 11.

5. Conclusion

Mobile ad hoc networks are formed by wireless devices without a predefined infrastructure. Due to highly dynamic environments, forwarding the data from a source to a destination is a critical and difficult task. This data routing can be treated using several techniques, but we rely on the bioinspired algorithms to solve this complex problem. Ant Colony Optimization (ACO) algorithms are our inspiration. ACO takes into account the behavior of ants at the time of obtaining the food. This behavior is based on Swarm Intelligence, that is, considers the collective behavior of some animals, in our case ants. In this paper, we present HACOR, a hybrid ACO-based routing protocol which has novel characteristics and employs enhanced techniques compared to its predecessors. The experimentation results show that HACOR has a better performance than AODV according to analyzed metrics. Overall, we can appreciate a significant improvement with regard to overhead in number of packets.

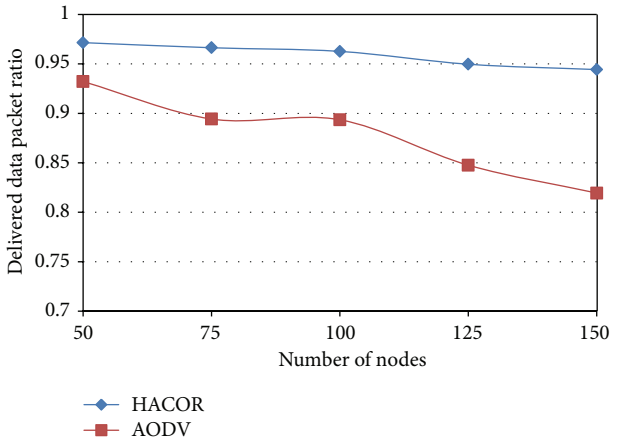


Figure 11: Delivered data packet ratio (Experiment B).

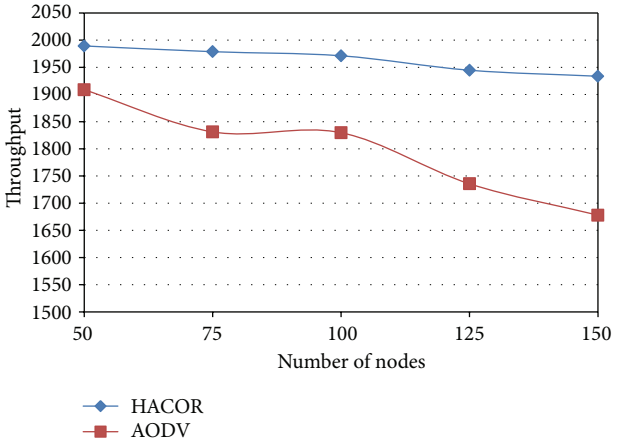


Figure 12: Throughput (Experiment B).

Acknowledgment

This work was supported by the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID MEDITERRÁNEO A1/037528/11.

References

[1] R. Ramanathan and J. Redi, "A brief overview of ad hoc networks: challenges and directions," *Communications Magazine, IEEE*, vol. 40, no. 5, pp. 20–22, 2002.

[2] L. J. G. Villalba, J. G. Matesanz, A. L. S. Orozco, and J. D. M. Díaz, "Auto-configuration protocols in mobile ad hoc networks," *Sensors*, vol. 11, no. 4, pp. 3652–3666, 2011.

[3] L. J. G. Villalba, A. L. S. Orozco, A. T. Cabrera, and C. J. B. Abbas, "Routing protocols in wireless sensor networks," *Sensors*, vol. 9, no. 11, pp. 8399–8421, 2009.

[4] L. J. G. Villalba, J. G. Matesanz, A. L. S. Orozco, and J. D. M. Daz, "Distributed dynamic host configuration protocol (D2HCP)," *Sensors*, vol. 11, no. 4, pp. 4438–4461, 2011.

[5] T. Clausen and P. Jacquet, *Optimized Link State Routing Protocol*, vol. 3626, RFC, 2003.

[6] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pp. 90–100, February 1999.

[7] M. Dorigo, "Ant colony optimization: a new meta-heuristic," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '99)*, pp. 1470–1477, IEEE Press, 1999.

[8] G. A. Di Caro, *Ant colony optimization and its application to adaptive routing in telecommunication networks [Ph.D. thesis]*, Faculte des Sciences Appliquees, Université Libre de Bruxelles, Brussels, Belgium, 2004.

[9] M. Dorigo, *Optimization, learning and natural algorithms [Ph.D. thesis]*, Politecnico di Milano, Italy, 1992.

[10] X. Zheng, W. Guo, and R. Liu, "An ant-based distributed routing algorithm for ad-hoc networks," in *Proceedings of the International Conference on Communications, Circuits and Systems (ICCCAS '04)*, vol. 1, pp. 412–417, June 2004.

[11] J. Wang, E. Osagie, P. Thulasiraman, and R. K. Thulasiram, "HOPNET: a hybrid ant colony optimization routing algorithm for mobile ad hoc network," *Ad Hoc Networks*, vol. 7, no. 4, pp. 690–705, 2009.

[12] M. K. Rafsanjani, S. Asadinia, and F. Pakzad, "A hybrid routing algorithm based on ant colony and ZHLS routing protocol for MANET," *Communications in Computer and Information Science*, vol. 120, no. 2, pp. 112–122, 2010.

[13] G. A. Di Caro, F. Ducatelle, and L. M. Gambardella, "AntHocNet: an antbased hybrid routing algorithm for mobile ad hoc networks," in *Proceedings of the of Parallel Problem Solving from Nature (PPSN '04)*, vol. 3242, pp. 461–470, Springer, 2004, Lecture Notes in Computer Science.

[14] F. Ducatelle, *Adaptive routing in ad hoc wireless multi-hop networks [Ph.D. thesis]*, Universita della Svizzera Italiana, Istituto Dalle Molle di Studi sull'Intelligenza, Lugano, Switzerland, 2007.

[15] L. J. G. Villalba, D. R. Cañas, and A. L. S. Orozco, "Bio-inspired routing protocol for mobile ad hoc networks," *IET Communications*, vol. 4, no. 18, pp. 2187–2195, 2010.

[16] D. R. Cañas, A. L. S. Orozco, L. J. G. Villalba, and T. H. Kim, "A comparison study between antor-disjointnode routing and antor-disjoint link routing for mobile ad hocnetworks," in *Proceedings of the FGIT-MulGraB*, pp. 300–304, 2011.

[17] G. F. Riley and T. R. Henderson, *The Ns-3 Network Simulator*, 2010.

Routing Techniques Based on Swarm Intelligence

Delfín Rupérez Cañas, Ana Lucila Sandoval Orozco and Luis Javier García Villalba *

Group of Analysis, Security and Systems (GASS)
Department of Software Engineering and Artificial Intelligence (DISIA)
School of Computer Science, Office 431
Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n
Ciudad Universitaria, 28040 Madrid, Spain
{delfinrc,asandoval,javiergv}@fdi.ucm.es

Abstract. Artificial Immune Systems (AIS) are used for solving complex optimization problems and can be applied to the detection of misbehaviors, such as a fault tolerant. We present novel techniques for the routing optimization from the perspective of the artificial Immunology theory. We discussed the bioinspired protocol AntOR and analyze its new enhancements. This ACO protocol based on swarm intelligence takes into account the behavior of the ants at the time of obtaining the food. In the simulation results we compare it with the reactive protocol AODV observing how our proposal improves it according to the delivered data packet ratio and overhead in number of packets metrics.

Keywords: Ant Colony Optimization, Artificial Immune System,Bioinspired Protocol, Mobile Ad Hoc Networks, Routing.

1 Introduction

Optimization problems can be solved by artificial immune systems. These problems we face with these kinds of problems daily: the efficiency improvement of the resources of the devices, find the shortest path between two points, distribute the resources in the system uniformly.

One of the optimization algorithms based on the colony of ants [1] and that relies on the intelligence swarm [2], has been frequently cited in the literature. It is inspired by the behavior of ants at the time of obtaining the food and in many areas is applied.

ACO algorithms are composed by agents that work without the need of a centralized control structure, in such a way that the interactions local of each an agent and its neighbors allow between them to communication in an autonomous way. These algorithms can be used to resolve routing problems, being suitable

* Corresponding author.

for highly dynamic environments. We present improvements in the optimization of protocol AntOR [3], in its disjoint-link version and we show its relationship with the artificial immune systems. A work related to immune systems is [4]. In this work the authors try to solve problems of misbehaviors in mobile ad hoc networks (MANETs) taking into account the artificial immune systems, but they have used the standard protocol DSR which it is reactive and it does not exploit the properties of the hybrids.

We structure the rest of article as follows: in section 2 we explain our proposal as a view point of immunology. In section 3 the simulation results in a dynamic environ-ment are exposed comparing them with the standard protocol AODV. Section 4 pre-sents conclusions.

2 Proposed Algorithm

We present the hybrid (mix between reactive and proactive part) routing protocol AntOR [3] with the following characteristics:

- Disjoint-link and disjoint-node protocol [5].
- Separation between the pheromones values in the diffusion process.
- Use of the distance metric in the proactive path exploration.

AntOR provides two versions in its design: the disjoint-link (AntOR-DLR) in which the links are not shared and disjoint-node (AntOR-DNR) in which the nodes are not shared. Every disjoint-node is also a disjoint-link, but not vice versa. Both types of disjoint routes have the following advantages:

1. A failure in one node only affects a path, not the entire network.
2. Load balancing is better because there are not repeated routes on the disjoint property.

However, the use of such routes needs more resources by not sharing the links or nodes.

This algorithm is applied to mobile ad hoc networks (MANETs) and the optimization might be addressed by ideas of immunology. This algorithm is modeled in the following way:

- **Body**: The entire mobile ad hoc network.
- **Antibody**: Address pairs consisting of the "next hop'" and "destination".
- **Antigen**: Destination of the data packet.
- **Matching**: Correspondence between the associated destination with the data packet and destination field of a pair which it belongs to an antibody.
- **Affinity**: Heuristic value (Regular pheromone).

The new techniques used and reflected in the simulation results are: reduction of overload of the system through proactive agents that do not need virtual pheromone routes. These agents create alternative routes and go from neighbor

to neighbor until reaching the destination node. At the time of selecting the next hop they take into account the maximum value of regular pheromone to such a one-hop neighbor. Alternative routes are achieved with this technique up to a limit which is selected previously, and which are disjoint because those routes do not belong to the main route.

Another technique is related to fault tolerant. When a fault in a control message (an agent of our algorithm) is detected, we trigger a mechanism of neutralization process. This makes that, in highly dynamic environments, we have to trigger more neutralization mechanisms by sending agents to repair the route or notify to the precursors of the node that detects the failure until reaching the source of the data session indicating that the route is disconnected. This implies an overhead in packets and bytes. To fix this we use a new technique that checks if exists route (regular pheromone value greater than zero) to the neighbor whose we want to transmit, seeing this information in the routing table. If path exists, we send the control message, otherwise the agent does not send. Thus, this prevents the failure neutralization and it reduces overhead.

3 Simulation Results

Performance metric are delivered data packet ratio and overhead in packets.

The characteristics of the simulations in network simulator NS-3 were: We used 100 nodes randomly distributed and configured with a transmission range of 300 m. The nodes are moved according to the *Random WayPoint* (RWP) pattern, varying pause time from a low of 0 s to 240 s at intervals of 60 s. The scenario was rectangular with dimensions 3000 m 1000 m. The speed was variable from a minimum of 0 m/s to 8 m/s. It used 10 random data sessions using the application protocol *Constant Bit Rate* (CBR) beginning to send data at random from 0 s to a maximum of 60 s. The sending rate was 512 bit/s, i.e., sending a packet of 64 bytes per second. The maximum simulation time was established to 300 s. It employed a total of 10 runs in the experiment.

Fig 1 shows how the data packet ratio in our proposed protocol AntOR-DLR is better than in AODV at all time. The ratio is an important metric of effectiveness.

Fig. 2 shows how the overhead in AntOR is practically the same as AODV.

4 Conclusion

In this article we have presented an ACO bioinspired algorithm and provided new optimization techniques, relating it with concepts of artificial immune systems. Finally we can see that these new techniques: reduction of the overhead in the system and fault tolerance originates this version AntOR-DLR to behave better than AODV according to metrics of ratio and overhead in number of packets.

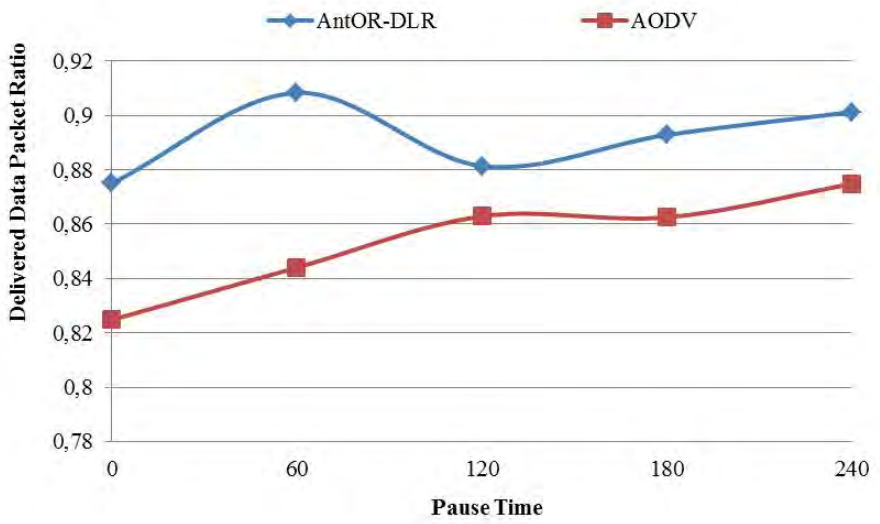


Fig. 1. Pause Time versus Ratio

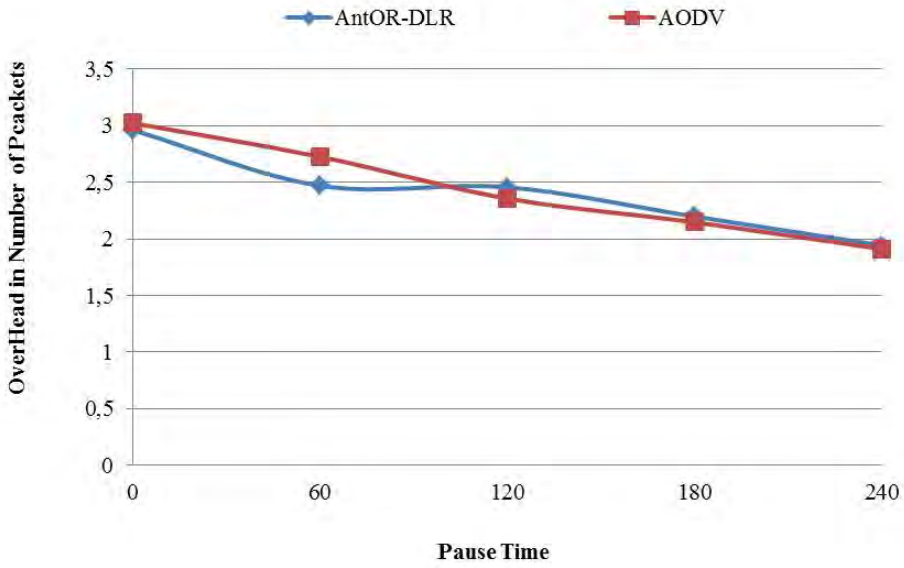


Fig. 2. Pause Time versus Overhead in packets

Acknowledgment

This work was supported by the Ministerio de Industria, Turismo y Comercio (MITyC, Spain) through the Project Avanza Competitividad I+D+I TSI-020100-2011-165 and the Agencia Española de Cooperación Internacional para el Desarrollo (AECID, Spain) through Acción Integrada MAEC-AECID MEDITERRÁNEO A1/037528/11.

References

1. M. Dorigo: Optimization, Learning and Natural Algorithms, Doctoral Thesis. Politecnico di Milano, Italie (1992).
2. J. Kennedy: Swarm Intelligence. Morgan Kaufmann Publishers (2001).
3. L. J. García Villalba, D. Rupérez Cañas, and A. L. Sandoval Orozco: Bioinspired routing protocol for mobile ad hoc networks, IET Communications, vol. 4, no. 18, pp. 2187-2195 (2010)
4. J. Le Boudec and S. Sarajanoviè: An Artificial Immune System Approach to Misbehavior Detection in Mobile Ad-Hoc Networks. In Proceedings of the First International Workshop on Biologically Inspired Approaches to Advanced Information Technology (Bio-ADIT 2004), Lausanne, Switzerland, January 29-30, pp. 96-111 (2004).
5. D. Rupérez Cañas, A. L. Sandoval Orozco, L. J. García and T.-H. Kim: A Comparison Study between AntOR-Disjoint Node Routing and AntOR-Disjoint Link Routing for Mobile Ad Hoc Networks, in Communications in Computer and Information Science (CCIS), vol. 263, pp. 300-304 (2011).